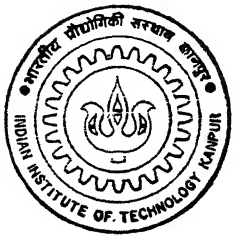
CONTINUOUS CASTING MOLD DESIGN USING GENETIC ALGORITHMS AND OTHER OPTIMIZATION TECHNIQUES

by

AMRITA MUKHERJEE



DEPARTMENT OF MATERIALS AND METALLURGICAL ENGINEERING

Indian Institute of Technology, Kanpur

DECEMBER, 1998

CONTINUOUS CASTING MOLD DESIGN USING GENETIC ALGORITHMS AND OTHER OPTIMIZATION TECHNIQUES

A Thesis Submitted

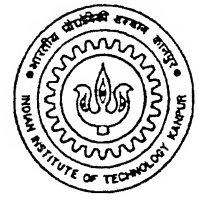
in Partial Fulfillment of the Requirements

for the Degree of

Master of Technology

by

AMRITA MUKHERJEE



to the

DEPARTMENT OF MATERIALS AND METALLURGICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

DECEMBER 1998

CERTIFICATE

It is certified that the work contained in this thesis entitled " *Continuous Casting Mold Design Using Genetic Algorithms and Other Optimization Techniques*", by *Amrita Mukherjee*, has been carried out under my supervision and that this work has not been submitted elsewhere for any degree.

Dr. N. Chakraborti
Professor
Materials and Metallurgical Engg.
Indian Institute of Technology, Kanpur
December, 1998

Dedicated to

My Parents

Acknowledgement

I wish to express my gratitude to Prof. N. Chakraborti for introducing me in the field of evolutionary computation. He taught me many things as his M.Tech student and uplifted my research ability from its infancy to the present state. The successful completion of this work has been possible only due his excellent guidance, intellectual support, meticulous observation and critical analysis. I thank him once again for his great patience in converting my vague ideas to a meaningful creation during thesis preparation.

I am also grateful to Prof. Kalyanmoy Deb for giving me the valuable suggestions and lessons in evolutionary computation.

My sincere thanks to Prof. D. Mazumdar for his immense help in uplifting my knowledge in the field of transport phenomena.

I would like to thank Mr. A. Saha of Materials Science Programme, Mr. G. Chakraborti of Mechanical Engineering Programme, and other seniors and friends in the hostel for providing me a cordial, friendly environment during my stay in IITK.

The unbreakable patience of my parents and their support were always a source of inspiration for me.

Abstract

In steel-making continuous casting is one of the widely used process used for casting billets, blooms and other semi-finished products. The mold region of a continuous caster plays a crucial role in determining the overall efficiency of the process. The solidified shell that develops in the mold region directly influences all the other operations downstream and becomes a major factor for the attainment of required structures and mechanical properties in the semi-finished products.

For proper understanding of the primary solidification process that occurs in the mold region, a heat transfer based optimization analysis of the mold has been carried out in this study to evaluate the maximum casting velocities under various conditions. To achieve the optimum values of the operational variables in the mold design problem, an emerging methodology in evolutionary computation, called *genetic algorithms*, coupled with other traditional and non-traditional optimization techniques has been adopted. In order to consider the physical phenomena of solidification process taking place in the mold, several objective functions were constructed, expressing casting velocity as a function of a total of eighteen process variables. A steady state heat balance along with the pertinent equations for mold oscillation and slag vitrification was used for this purpose. A number of constraints arising out of physical considerations were also considered.

The reliability of the model proposed in this study was verified by comparing the results obtained by this rigorous optimization scheme with a significant amount of industrial data available in the literature. A functional relationship of the optimum casting speed with

design variables was found by optimizing the parameters within predefined ranges. The efficiency, search power and complexity of different optimization techniques such as *genetic algorithms*, *simulated annealing* and *differential evolution* had also been compared in this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Continuous casting process is one of the major technological developments in the 20th century which is gradually replacing the ingot casting operation for the production of the semi-finished blooms, billets and slabs. Only a brief introduction is necessary to comprehend the continuous casting revolution that is shaping up the steel-making process, as a tremendous amount of research effort has gone into this particular area in recent years and many detailed reports are available in the literature[1, 2, 3]. The continuous casting involves a sequence of steps which is shown schematically in Figure 1.1. Liquid steel is first transferred from a steel-making ladle to a tundish. The tundish holds the steel during casting process and enables control of steel flow into a mold [4].

The mold region of a continuous caster is known to play a very crucial role in the overall efficiency of the process [3]. In this region primary cooling of liquid steel takes place. Therefore, formation of a solidified shell on the slab or billet surface is initiated in this region. The process of solidification continues as the billet or slab passes downward in the caster guided by the support rolls. The slab or billet is then additionally cooled by a series of water sprays located in the secondary cooling region which includes support rolls enclosed in a roller apron. The whole solidification process comes to an end after passing through this region. Finally, the product comes down through the withdrawal rolls. Later, the billet or

slab passes through the bending and straightening rolls and is cut into pieces of predefined length.

Semi-finished steel, produced by continuous casting process, is generally of acceptable quality in terms of its cleanliness, cracks, segregation and shape. Furthermore, the inherent advantages of low-cost, high yield, and flexible operation along with the ability to achieve a high quality cast product [5] has been increasingly prompting the steel industries worldwide to adopt the continuous casting technology for the last two decades. The continuous cast steel can directly be rolled or hot charged to a re-heating furnace without cooling for an intermediate inspection which adds to the efficiency of this process. Casting of good quality products through this route however requires a proper understanding of the key operations of the caster that influence the operator's ability to control and monitor the process effectively. The key factors those need to be studied are *fluid flow, heat flow, mechanical properties of the steel at the elevated temperature, stress-generation and solidification* [6].

The fluid flow in the tundish and mold has a strong influence on the cleanliness of the steel as it is responsible for the maximum inclusion float-outs. In addition production of finished products with low inclusions content requires control of the following aspects [7]:

(i) delivery of clean steel from tundish to caster.

(ii) minimization of oxygen transfer from air and the refractories.

(iii) prevention of the exogenous inclusion pickup from the refractories, ladle, tundish or mold powder.

(iv) usage of optimum mold powder.

The extraction of heat in mold and spray cooling zone is the major operation taking place in a continuous caster. In order to predict the temperature field in the solidifying steel and in the mold wall several mathematical models have been proposed so far. The

models are able to simulate the casting operations depending upon the actual measurement of surface heat flux in the different cooling regions [8].

In the mold, heat is transferred from the liquid steel to the cooling water via an air gap which separates the mold and the strand, the mold wall, and the interface between the outer mold wall and the cooling water. Among these, the air-gap provides the largest resistance to heat flow. In fact, the amount of heat transferred is almost inversely proportional to the air gap width [5, 6]. However the air gap width depends in a complex manner on the operational variables, and is almost impossible to predict accurately using a mathematical expression. The air gap width also varies both in longitudinal and transverse directions, resulting in a non-uniform heat extraction pattern. Heat transfer affects various factors of the steel making process. Among the factors which are influenced by heat extraction from the mold the following are of importance [9].

(i) For a particular steel composition the extent of thermal shrinkage and the phase transformation shrinkage is governed by the temperature distribution of the solidified shell. This in turn is affected by the mold heat transfer.

(ii) The ability of the shrinking solidified shell to resist the ferrostatic pressure depends on the evolution of the gap. The gap evolution is affected by the mold wall temperature, the solidified shell temperature and its thickness, and the mold heat extraction [5, 10].

(iii) The mold shape, especially the taper of the billet mold, is one of the important factors influenced by the heat transfer process. The narrow face of the slab mold and the walls of a billet mold are tapered to compensate for shrinkage. The billet molds may distort because of the differential thermal expansion coupled with the lack of constraints. The resultant change in taper may cause oscillation marks to form on the billet surface which in turn tend to enlarge the strand-mold gap width. Slab molds are less prone to this problem [10].

All these factors make computation of mold heat transfer from the first principle very difficult. Among them, control of the oscillation marks is one of the prime requirements for quality casting. In addition to the heat transfer process oscillation marks also depend on oscillation stroke and frequency, lubrication properties of oil or mold powder, magnitude of taper at the meniscus level, and all other variables affecting the mold distortion. In billet casting with oil lubrication, oscillation marks occur by mechanical interaction of mold with the billet during the downward stroke of oscillation cycle. In slab casting with mold powder this occurs because of the pressure generated in the mold flux channel between the mold and strand during the down stroke of oscillation [5, 6].

It follows from the above discussion that in continuous casting of steel, control of both fluid flow and heat transfer are very crucial. This is also very important in order to achieve the product quality and associated productivity [8]. An analysis of the solidification phenomena under continuous casting conditions also helps in understanding the segregation problems and its remedies. The solidification pattern that develops in the mold region directly influences all the operations downstream, and becomes a key factor in the attainment of the required texture in the finished products [11]. The studies relating to continuous casting mold are rather limited to the traditional heat-transfer mechanisms. The rigorous research works of Brimacombe et al. [5] have achieved what perhaps could easily be achieved through the numerical solutions of the heat-transfer equations obtained by a very sophisticated experimentation scheme [3]. In order to achieve the better understanding, one at this stage needs an approach drastically different from what has been done so far. In this work, we are going to elaborate one such methodology for tackling the continuous casting mold design problem. In order to do this we have adopted an optimization scheme of the decision variables using *Genetic Algorithms* (GA), an emerging methodology in evolutionary computation [3] which has been tested against a number of other optimization techniques. Till now, only a few applications of GA are reported in the field of process metallurgy. However,

it is likely to influence the future research in this area quite significantly.

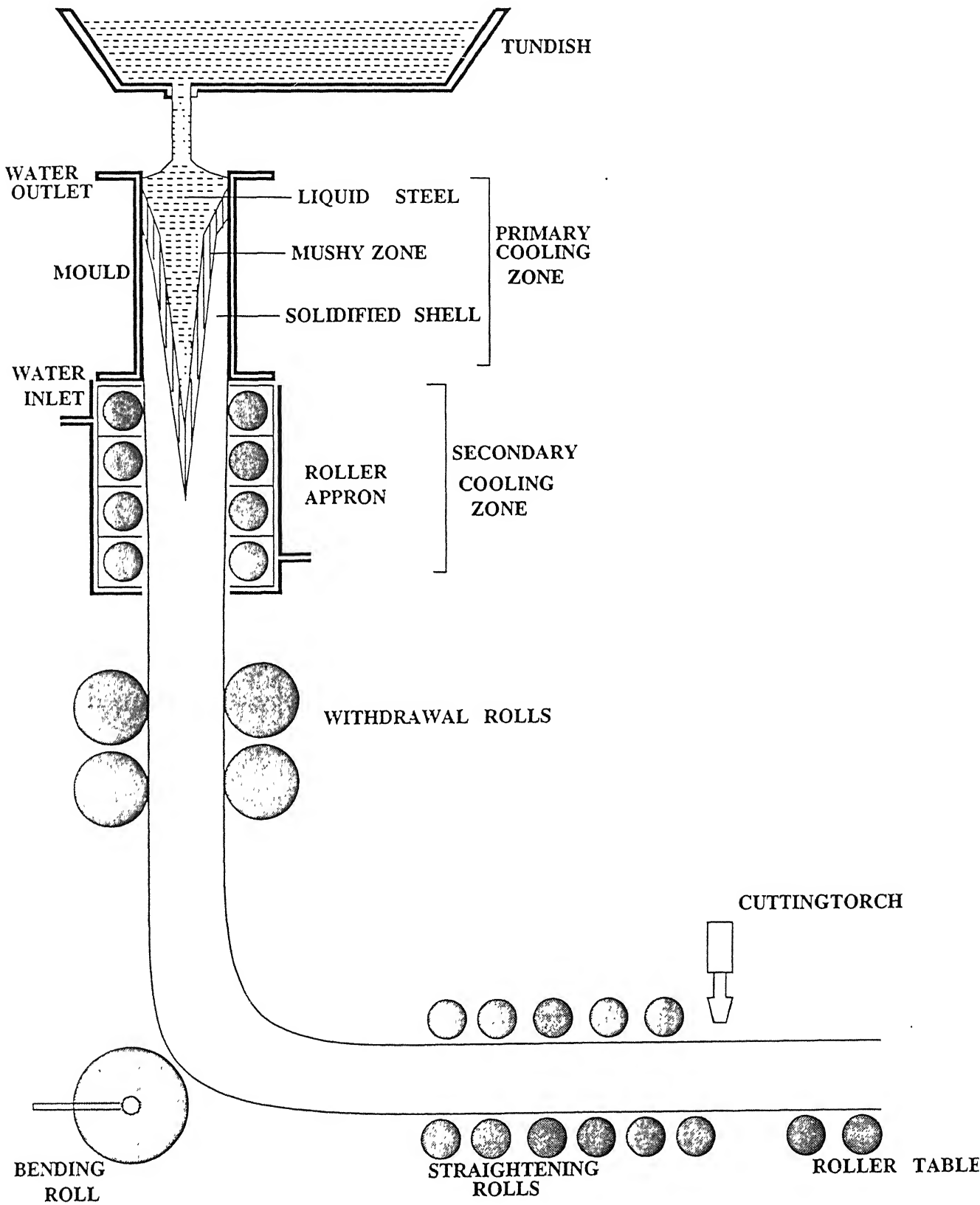


Figure 1.1: Basic operations of a continuous casting machine

Chapter 2

Fundamentals of Optimization Methods

2.1 Traditional Methods

An engineering design or decision making problem has an objective of minimizing or maximizing a function and simultaneously have a requirement for satisfying some constraints arising due to space, strength, or stability considerations. A constrained optimization problem comprises of an objective function together with a number of equality or inequality constraints. Often lower bound or upper bounds on design or decision variables are also specified. Considering N design or decision variables a single-objective constrained optimization problem is formulated in the following fashion [15].

$$\begin{aligned} &\text{Minimize or Maximize} && f(X) \text{ , where } X = [x_i] \\ &\text{Subject to} \\ &\left.\begin{aligned} & g_j(X) \geq 0 && j = 1, 2, ..J \\ & h_l(X) = 0 && l = 1, 2...L \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)} && where, i = 1, 2, ...N \end{aligned}\right\} \end{aligned} \tag{2.1}$$

Where, the function $f(X)$ is the objective function which is to be optimized. The functions $g_j(X)$ and $h_l(X)$ are inequality and equality constraints, respectively. In the above formulation, J such inequality constraints and L such equality constraints have to be satisfied.

Any point $X^{(t)}$ *satisfies* a constraints, if the left side expression of the constraint at that point agrees with the right-side value by the relational operator between them. A point(or solution) is defined as a *feasible* point (or solution) if all the equality and inequality constraints and the variable bounds are satisfied at that point. All the other points are known as *infeasible* points.

2.1.1 FUNDAMENTALS OF THE PENALTY FUNCTION METHOD

Penalty function methods transform the basic optimization problem in such a way that numerical solutions are obtained by solving a sequence of unconstrained optimization problems [12]. This alternative unconstrained formulation is done by adding penalty terms for each constraint violation. If a constraint is violated at any point, the objective function is penalized by an amount depending upon the extent of constraint violation. The penalty terms vary in the way the penalty is assigned [12].

Some penalty methods cannot deal with infeasible points at all and even penalize feasible points which are close to the constraint boundary. These methods are known as *Interior penalty* methods. In these methods, every sequence of the unconstrained optimization finds a feasible solution. The other kind of the penalty methods penalize *infeasible* points but do not penalize *feasible* points. These methods are known as *Exterior penalty* methods. In these methods, every sequence of the unconstrained optimization finds an improved yet infeasible solution [15]. If the optimum solution is an interior point in the feasible region, one sequence of the exterior penalty method should find the optimum point. On the other hand, if the optimum point is on a constraint boundary, several sequences may be necessary.

The optimization problem described above, is converted into an unconstrained opti-

mization problem by constructing a function of the form [12]

$$\phi_k = \phi(X, r_k) = f(X) + r_k \sum_{j=1}^{J} G_j[g_j(X)] \tag{2.2}$$

where G_j is some function of the constraint g_j, and r_k is a positive constant known as the penalty parameter. The second term on the right hand side of the Equation (2.2) is known as the penalty term. If the unconstrained optimization of the ϕ-function is repeated for a sequence of values of the penalty parameter $r_k, (k = 1, 2, ...N)$, the solution may be brought to convergence for the original problem stated by Equation (2.1) [12]. Thus, the penalty function methods are also known as *sequential unconstrained optimization* techniques.

In the interior formulation, some of the popularly used form of G_j are given by [12]

$$\text{Inverse Penalty: } G_j = \{\frac{1}{g_j(X)}\}$$

$$\text{Log Penalty: } G_j = -ln\{g_j(X)\}$$

In case of the exterior penalty function formulation the commonly used forms of the function G_j are [15]

$$\text{Infinite Barrier Penalty: } G_j = \left[\sum_{j \in \bar{J}} |g_j(X)|\right]$$

where $\bar{J}$ denotes the set of violated constraints at the current point.

$$\text{Parabolic Penalty: } G_j = max\{0, g_j(X)\}^2$$

The convergence of the unconstrained minimization of ϕ_k is illustrated for the simple problem, given below [12].

Find $X = x_1$ which minimizes $f(x) = \alpha x_1$

subject to

$$g_1(X) = x_1 - \beta \geq 0$$

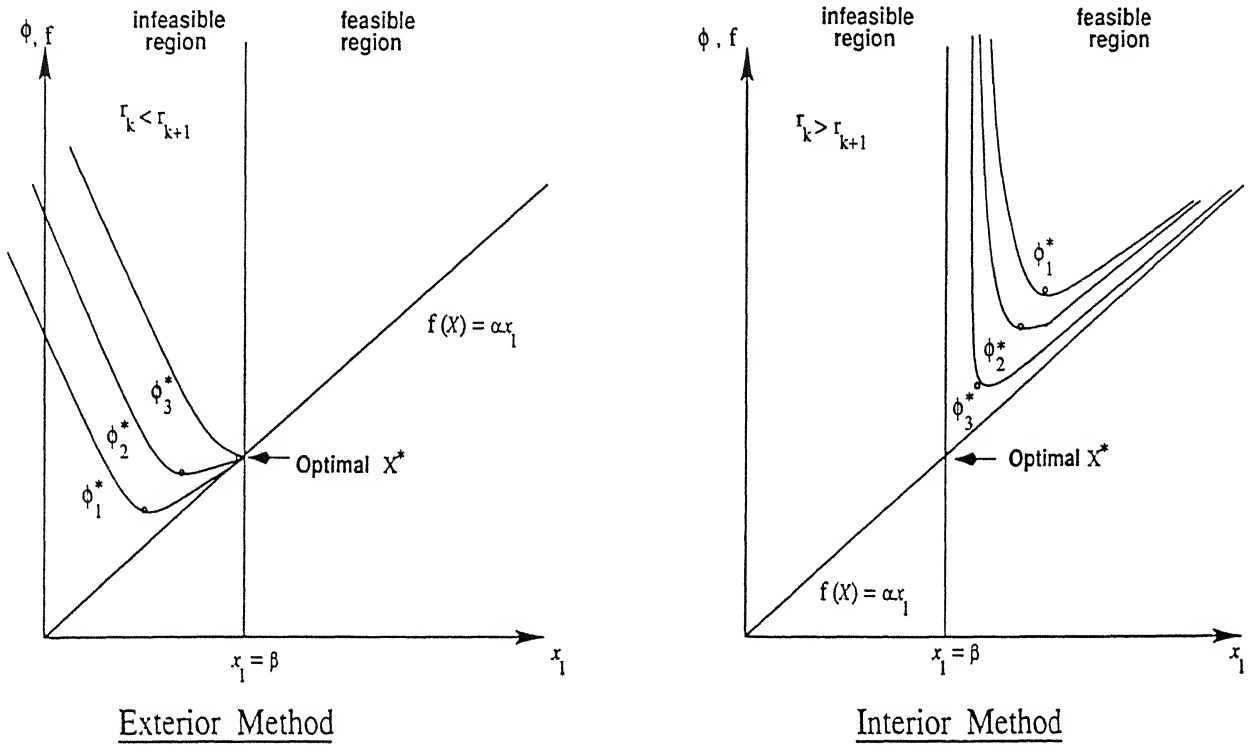


Figure 2.1: The gradual convergence of the Penalty Function Method

It can be seen from the Figure 2.1 that the unconstrained minima of the function $\phi(X, r_k)$ converge to the optimum point X^* as the parameter r_k is increased sequentially for the exterior method. On the other hand, the interior method shown in the figure gives convergence as the parameter r_k is decreased sequentially [12].

The sequential nature of penalty function method allows a gradual approach towards the *criticality* of the constraints [12]. In addition, the sequential process permits a graded approximation, which can used in the analysis of the system. This means that if the evaluation of f and $g_j(X)$ [hence $\phi(X, r_k)$] for any specified design vector X is computationally very difficult, we can use coarse approximation during the early stages of optimization, and finer or more detailed analysis and approximation during the final stages of optimization.

In the exterior penalty function method, the ϕ-function is generally taken as

$$\phi(X, r_k) = f(X) + r_k \sum_{j=1}^{J} \langle g_j(X) \rangle^q \tag{2.3}$$

where, r_k is the positive penalty parameter, the exponent q is nonnegative constant. The bracket function $\langle g_j(X) \rangle$ is defined as

$$\langle g_j(X) \rangle = max\langle g_j(X), 0 \rangle = \begin{cases} 0, & if g_j(X) \geq 0 \\ & \text{(constraint is satisfied)} \\ g_j(X), & if g_j(X) < 0 \\ & \text{(constraint is violated)} \end{cases}$$

It can be seen from Equation(2.3),that the effect of the second term on the right hand side is to increase $\phi(X, r_k)$ in proportion to the q-th power of the amount by which the constraint is violated [12]. Usually,the function $\phi(X, r_k)$ possesses a minimum as a function of X in the infeasible region. The unconstrained minima X_k^* converge to the optimal solution of the original problem, as $k \to \infty$ and $r_k \to \infty$ Thus the unconstrained minima approach the feasible domain gradually, and as $k \to \infty$, (X_k^*) eventually lies in the feasible region.

For $q > 1$, the ϕ-function will have continuous first derivatives, which is given by

$$\frac{\partial \phi}{\partial x_i} = \frac{\partial f}{\partial x_i} + r_k \sum_{j=1}^{J} q \langle g_j(X) \rangle^{q-1} \frac{\partial g_j(X)}{\partial x_i}$$

Generally, the value of q is chosen as 2 in practical computation.

The *ALGORITHM* for the *Exterior Penalty Function Method* is described in the following steps.

(i) Start from any design vector X_1 and suitable value of r_1, set K=1,

(ii) Find the vector X_k^* that minimizes the function

$$\phi(X, r_k) = f(X) + r_k \sum_{j=1}^{J} \langle g_j(X) \rangle^q$$

(iii) Test whether the point X_k^* satisfies all the constraints. If X_k^* is feasible, it is the desired optimum.Hence **Terminate**. otherwise go to step(iv).

(iv) Choose the next value for the penalty parameter which satisfies the relation $r_{k+1} > r_k$, i.e. $\frac{r_{k+1}}{r_k} = c > 1$. Set, K=K+1 , go to step(ii).

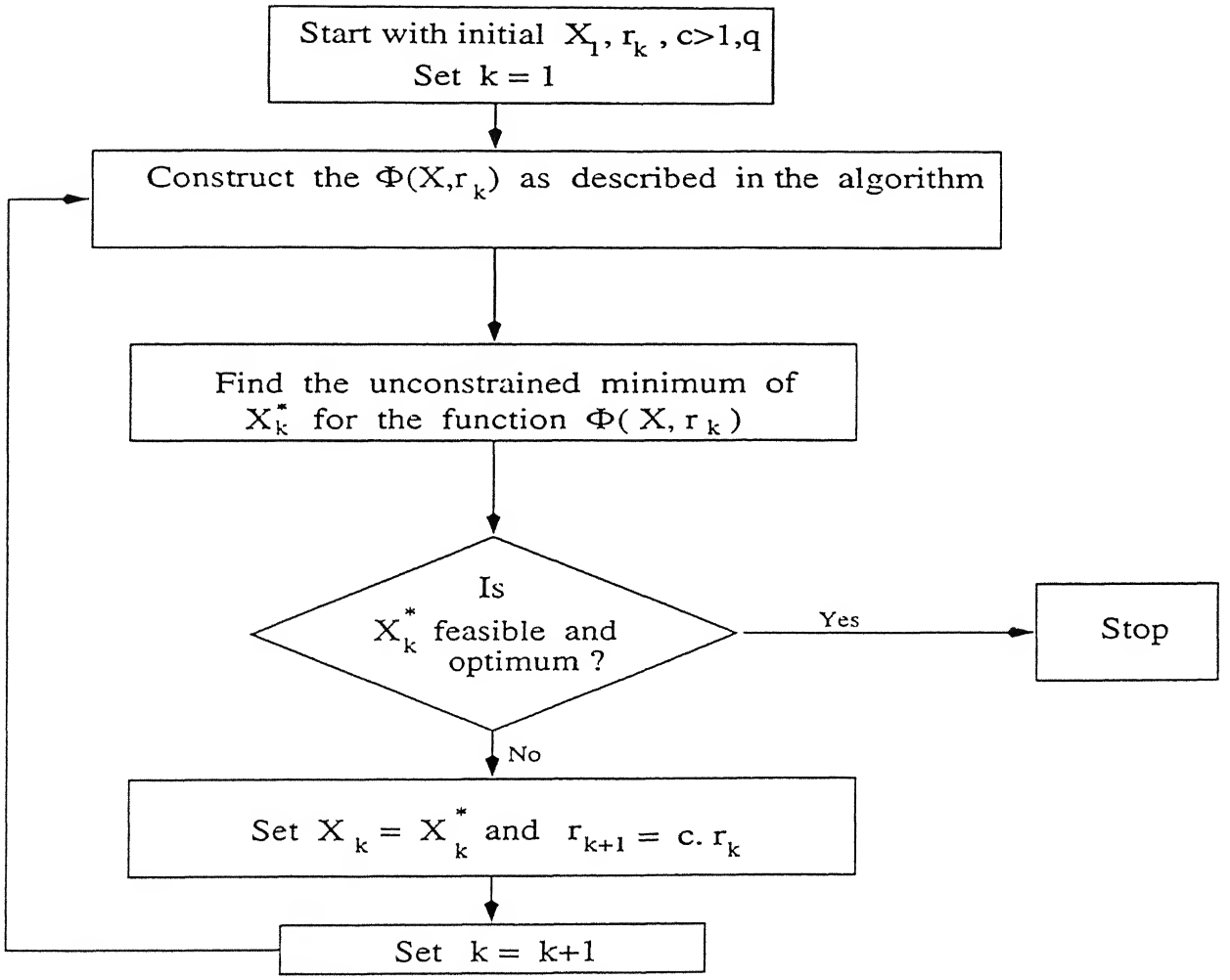


Figure 2.2: The flow chart for the Exterior Penalty Function Method

A flow-chart for this method is shown in Figure 2.2. To solve an optimization problem, involving both equality and inequality constraints the following form of ϕ-function has been

formulated.

$$\phi_k = \phi(X, r_k) = f(X) + r_k \sum_{j=1}^{J} \langle g_j(X) \rangle^q + r_k \sum_{l=1}^{L} h_k(X)^2 \tag{2.4}$$

The unconstrained ϕ-function can be minimized using **Steepest Descent Method**, as described below.

2.1.2 STEEPEST DESCENT METHOD

The search direction for minimization is the negative of the gradient vector of the function at any point X

$$S_i = -\nabla f_i(X) \tag{2.5}$$

Starting from an initial trial point X_1 the algorithm searches towards the optimum point according to the rule

$$X_{i+1} = X_i + \lambda_i^* S_i = X_i - \lambda_i^* \nabla f_i(X) \tag{2.6}$$

where λ_i^* is the optimal step length along the search direction S_i or $-\nabla f_i(X)$.

The *ALGORITHM* for the *Steepest Descent Method* is described in the following steps.

(i) Choose a maximum number of iterations M to be performed, an initial point X_0, two termination criteria ϵ_1 and ϵ_2. Set i = 0.

(ii) Calculate $\nabla f(X_i)$, the first derivative of the function at point (X_i)

(iii) If $||\nabla f(X_i)|| \leq \epsilon_1$, **Terminate.** Else if $i \geq M$, **Terminate.** Else go to step (iv).

(iv) Perform *Unidirectional Search* to find λ_i^* using ϵ_2 such that,

$$f(X_{i+1}) = f(X_i - \lambda_i^* \nabla f_i)$$

is minimum. Another Criteria for termination is when ,

$$|\nabla f(X_{i+1}).\nabla f(X_i)| \leq \epsilon_2$$

(v) Check, whether

$$\frac{||X_{i+1} - X_i||}{||X_i||} \leq \epsilon_1?$$

If yes, *Terminate.* Else set $i = (i + 1)$, and go to step (ii)

A flow chart for the method is shown in Figure 2.3. The *Steepest Descent* uses the *bounding*

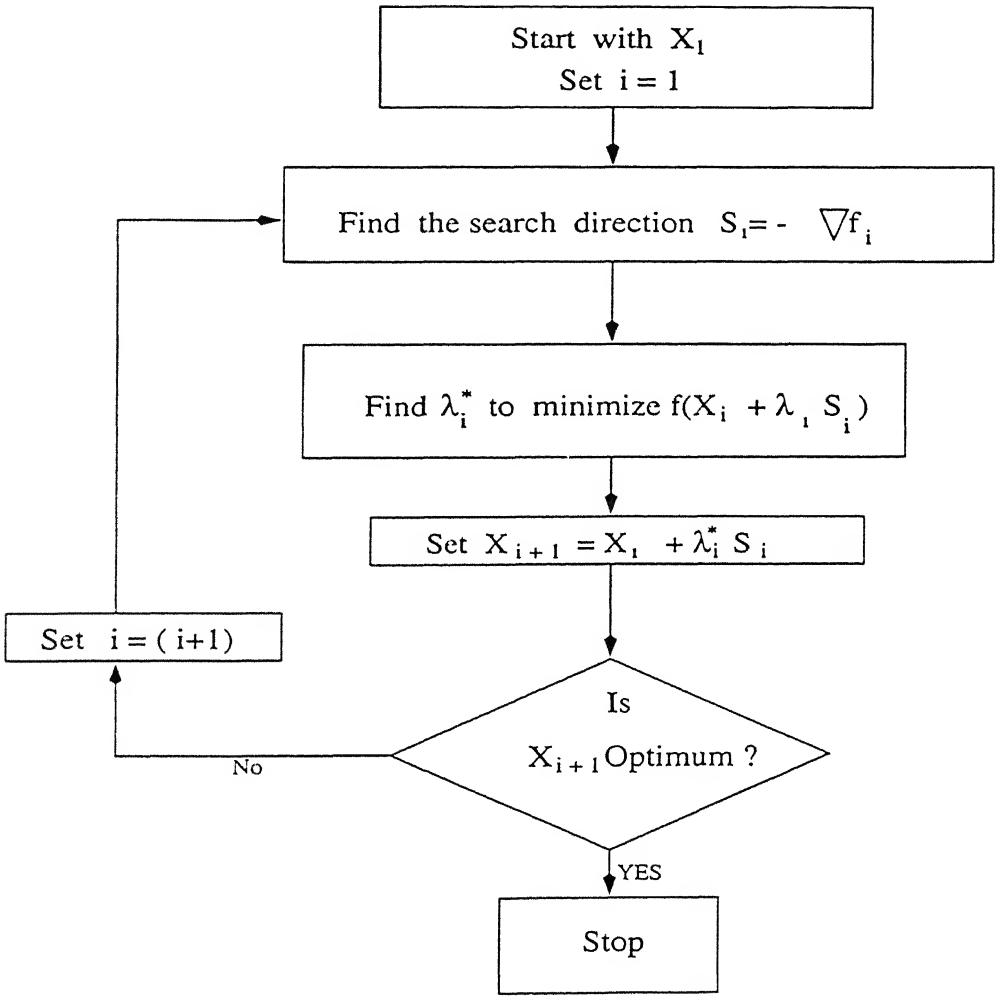


Figure 2.3: The flow chart for the Steepest Descent Method

phase and the *golden section* methods in order to perform the unidirectional search to find λ^* which minimizes $f(X_i + \lambda_i S_i)$, which are briefly described below.

2.1.3 BOUNDING PHASE METHOD

The *ALGORITHM* for the *Bounding Phase Method* involves the following steps.

(i) Choose an initial guess x_0 and an increment Δ. Set $k = 0$

(ii) If $f(x_0 - |\Delta|) \geq f(x_0) \geq f(x_0 + |\Delta|)$,then Δ is positive; Else if $f(x_0 - |\Delta|) \leq f(x_0) \leq f(x_0 + |\Delta|)$,then Δ is negative; Else go to step (i).

(iii) Set $x_{k+1} = x_k + 2^k\Delta$.

(iv) If $f(x_{k+1}) < f(x_k)$, set $k = (k+1)$ and go to step(iii). Else the minimum lies in the interval (x_{k-1}, x_{k+1}) and *Terminate*

2.1.4 GOLDEN SECTION METHOD

The *ALGORITHM* for the *Golden Section Method* is described in the following steps.

(i) Choose a lower bound a and an upper bound b . Also choose a small number ϵ . Normalize the variable x by using the equation $w = \frac{(x-a)}{(b-a)}$. Thus $a_w = 0, b_w = 1$, and $L_w = 1$. Set $k = 1$.

(ii) Set $w_1 = a_w + (0.618)L_w$ and $w_2 = b_w - (0.618)L_w$ Compute $f(w_1)$ or $f(w_2)$, depending upon whichever of the two was not evaluated earlier. Use the fundamental *Fundamental Region Elimination Rule* [15] to eliminate a region: Set new a_w and b_w.

(iii) Is $|L_w| < \epsilon$(a small number for termination)? If no , set $k = (k+1)$, go to step (ii), Else *Terminate*

2.2 FUNDAMENTALS OF THE GENETIC ALGORITHMS

Genetic Algorithms (GA) differ from the traditional search and optimization procedures in a numerous way. Basically, most of the traditional procedures are gradient-based, i.e.

starting from an initial guess vector, the search algorithm proceeds iteratively toward the optimal point taking the gradient information into account in the N-dimensional search space. In the previous section, one of the traditional method and its procedure has been described in details, showing how it leads to nearby optimum. In *genetic algorithms*, the approach is entirely different, as it work with the coding of the design variables, called *individuals*, a matter of which is very uncommon in traditional methods. This allows *genetic algorithms* to be applied to discrete or discontinuous function optimization problems. Since no gradient information is used in GAs, they can also be applied to non-differentiable function optimizations. This makes *genetic algorithms* much more robust in the sense that they can be applied to a wide variety of problems.

Unlike, many traditional optimization methods, *genetic algorithms* work with a *population* of points, which more or less ensures the possibilities of obtaining the global optimal solution for any given problem. *Genetic Algorithms* use *the probabilistic transition rule* instead of fixed rules. The group of *individuals*, called *population* in *genetic algorithms* terminology, are guided to the global optimum, by powerful genetic operators, such as *Reproduction, Crossover and Mutation.* The randomness in the *genetic algorithms* operators has an effect of producing impartial search direction early on, thereby avoiding a hasty wrong decision values; GAs have a directed search later in the simulation.

Genetic algorithms are search procedures which are based on the mechanism of natural genetics and natural selection. The basic principle of *genetic algorithm* is based upon the quasi-Darwinian principle of survival of the fittest, combined with simulated genetic operators abstracted from nature, where an individual attempts to pass its clone to the next generation and it is evaluated in terms of its *fitness* which is either the function itself or some appropriate transformation of it [17]. For example, in a maximization problem of a function $\phi(\mathrm{X})$ in between the upper and lower bounds of the decision variable $x_i^{(L)} \leq x_i \leq x_i^{(U)}$, the fitness function *f(X)* is defined same as $\phi(\mathrm{X})$, but on the contrary, for a minimization

problem the fitness function may be defined as,

$$f(X) = \frac{1}{1+\phi(X)} \tag{2.7}$$

Several other suitable transformation methods are also available depending upon the nature and complexity of the problem. These transformation methods do not alter the location of the global optimum, but convert the minimization problem into a maximization problem.

For any search and learning methods the way in which candidate solutions are encoded is a central factor in the success of a *genetic algorithms* [17]. In most *genetic algorithms* applications, *fixed-length,fixed-order bit strings* are used to encode candidate solutions [17]. The decision variables are usually mapped into the desired interval in the corresponding solution space by a *string(chromosome)* of binary alphabets '1's and '0's*(gene)*. The length of the string depends upon the precision of the desired solution. In a multi-parameter optimization problem, individual parameter codings *(substrings)* are concatenated together to form a bigger complete string. Eventhough binary codings are mostly applied, but it is not absolutely necessary. There exist some studies, where the decision variables are encoded in real numbers.

For example, if the five bit string are used to code the variable x_i, then the strings (00000) and (11111) would represent $x_i^{(L)}$ and $x_i^{(U)}$, since those substrings have the minimum and maximum decoded value. For converting the binary substrings to real space, linear mapping rule is common among the several transformation schemes [15].

$$x_i = x_i^{(L)} + \frac{x_i^{(U)} - x_i^{(L)}}{2^{l_i} - 1}(S_i) \tag{2.8}$$

where the variable x_i is coded in a substring S_i of length l_i. The decoded value of a binary substring S_i is calculated as

$$S_i = \sum_{i=0}^{l-1} 2^i(s_i) \tag{2.9}$$

where $s_i \in (0,1)$. For five bits to code each variable, there are only 2^5 or 32 distinct substrings possible, because each bit position can take either 0 or 1.

First, a *population* of strings representing the decision or design parameters is created at random. The size of the population depends upon the string length and the problem being optimized. Each string is then evaluated. The evaluation procedure first requires decoding of the decision variables from the chromosomes. Once the values of the decision variables are obtained, they are used to calculate the *fitness* function value. After that the *Reproduction Operator* takes over and *mating pool* is created. Further details are provided below.

2.2.1 REPRODUCTION OPERATOR

Reproduction is the first operator applied on a population. Reproduction selects good strings in a population and forms a *mating pool* [15]. The commonly used reproduction operator is the *Proportionate Reproduction*, where a string is selected for the mating pool with a probability proportional to its fitness value. Thus, the i-th string in a population is selected with a probability proportional to to $\phi(X)$. Since the population size is usually kept constant in a *Simple Genetic Algorithm* (SGA), thus the sum of the probability of each string being selected for the mating pool must be one. Thus, the probability for selecting the i-th string is [15]

$$P_i = \frac{\phi_i}{\sum_{j=1}^{n} \phi_j} \tag{2.10}$$

where, n is the population size.

The way to implement this selection operator, is known as *"Roulette Wheel Selection"* in GA literature [19].

In *"Roulette Wheel Selection"* scheme, a roulette wheel is imagined with a portion of its circumference marked for each string which is proportional to the string's fitness [15]. The roulette wheel is spun n times, and each time selecting an instance of the string chosen by the roulette wheel pointer. Since, the circumference of the wheel is marked according to the string's fitness, thus it can be expected to make $\frac{\phi_i}{\phi_{avg}}$ copies of the i-th string in the mating pool, when the *average fitness* of the population is calculated as [15]

$$\phi_{avg} = \frac{\sum_{i=1}^{n} \phi_i}{n} \tag{2.11}$$

In *"Stochastic Remainder Selection"*, the expected count for each individual is calculated. Then the strings are first assigned the number of copies exactly equal to the mantissa of the expected count. Thereafter, the regular *"Roulette Wheel selection"* is implemented using the decimal part of the expected count as the probability of selection. Furthermore *"Stochastic Remainder Selection"* selection method is less noisy as compared to *"Roulette Wheel Selection"* [15].

In *Simple Genetic Algorithm* (SGA), the population size is usually kept constant. Thus, the probability for selecting an *individual* from i-th class [1] as in the t-th generation is given by

$$p_{i,t} = \frac{f_i(x)}{\sum_{j=1}^{k} m_{j,t} f_j(x)} \tag{2.12}$$

where m denotes the number of individuals, in which k classes exist. According to Goldberg and Deb [20], for a non-overlapping population of constant size n, the expected number of copies of the i-th class in the next generation is

$$m_{i,t+1} = m_{i,t}.n.p_{i,t} = m_{i,t}.\frac{f_i}{\bar{f}_t} \tag{2.13}$$

where $\bar{f}_t$, the average function value of the current generation is given as

$$\bar{f}_t = \frac{\sum m_{i,t} f_i}{n}$$

This above equation may be written in proportional form dividing by the population size n as

$$P_{i,t+1} = P_{i,t}\frac{f_i}{\bar{f}_t} \tag{2.14}$$

By solving the Equation 2.12 explicitly, it can be shown that

$$P_{i,t} = \frac{f_i^t P_{i,0}}{\sum_j f_j^t P_{j,0}} \tag{2.15}$$

In order to calculate the performance measure and time complexity of the reproduction operators, Goldberg and Deb [20] have defined *take-over time* as the time during which

[1] Here, the *class* is essentially defined as individual's relationship with a particular form of *schema* [19] in the population.

the population contains $(n-1)$ best individuals out of n population size starting from the beginning. Considering, the monomial test case, the takeover time can be calculated as

$$t^* = \frac{1}{c}(n \ln n - 1) \tag{2.16}$$

Thus the takeover time for a polynomially distributed objective function is $O(n\ logn)$. For comparison, the takeover time for an exponentially distributed (or exponentially scaled) function is given by [20]

$$t^* = \frac{1}{c} n \ln n \tag{2.17}$$

Thus, under the unit interval consideration, both a polynomially distributed function and the exponentially distributed function have the same rate of convergence.

The major problem with *fitness proportionate selection* is that early in the search the fitness variance in the population is high and a small number of individuals are more fit than the others. Under *fitness proportionate selection* they and their *descendent* will multiply quickly in the population, in effect preventing *genetic algorithms* from doing any further *exploration*, causing *premature convergence* [17]. In this situation *Fitness proportionate selection* results in a lack of diversity in the population. Avoidance of such problem involves the usage of low *selection pressure* (the degree to which highly fit individuals have many offsprings [17]) at the initial period when the population variance is high and increase the pressure at a later time period when the variance decreases.

Rank Selection is the another selection operator, where sorting of the population is done from best to worst, the number of copies that each individual should receive is assigned according to a non-increasing assignment function and then a proportionate selection is performed according to that assignment. Ranking avoids giving the far largest share of offspring to a small group of highly fit individuals and reduces the selection pressure when the fitness-variance is high and keeps up the selection pressure when the fitness variance is low [17]. According to Baker [17], in the linear ranking method each individual in the population is ranked in increasing order of fitness from 1 to N. The expected value of each individual i in the population at time t is given by

$$ExpVal(i,t) = Min + (Max - Min)\frac{rank(i,t) - 1}{N - 1} \tag{2.18}$$

where Min is the expected value of the individual with *rank 1* and Max (this is assigned by the users) is the expected value of the individual with *rank N*. Goldberg and Deb [20] had shown that the takeover time for such a selection methodology is

$$t^* = \frac{2}{c} log(n-1) \tag{2.19}$$

This kind of selection technique had been identified to be potentially time consuming, especially when the population size is large.

In order to make the selection methodology in *genetic algorithms* less susceptible to *premature convergence*, *Sigma Scaling technique* has been adopted [17], which keeps the selection pressure relatively constant over the course of run rather than depending on the fitness variances in the population. Under *Sigma Scaling* an individual's expected value is a function of its *fitness, the population mean, and the population standard deviation.*

$$ExpVal(i,t) = \begin{cases} 1.0 + \frac{f(i)-\bar{f}(t)}{2\sigma(t)} & \text{if } \sigma(t) \neq 0 \\ 1.0 & \text{if } \sigma(t) = 0 \end{cases} \tag{2.20}$$

where *ExpVal(i,t)* is the expected value of the individual i at time t, $f(i)$ is the fitness of i, $\bar{f}(t)$ is the mean fitness of the population at time t, and $\sigma(t)$ is the standard deviation of the population fitnesses at time t.

Tournament Selection Operator is similar to rank selection in terms of selection pressure, but it is computationally more efficient and more amenable to parallel implementation. In this selection mechanism a number of individuals (generally two) are selected at random from the population which are subjected to compete with each other. An uniform random number r is generated between 0 and 1, if $r < k$ (where k is a parameter of preassigned value between 0 and 1), then the better of the two individuals is selected for the mating pool. Finally, both are returned to the original population and can be selected again. According to Goldberg and Deb [15, 19], the idea of tournament selection basically comprises of selecting some number of individuals randomly from a population (with or without replacement), selecting the best one from this group for further genetic processing, and repeating the sequences as often as desired (usually until the *mating pool* is filled). By solving for the takeover time yields an asymptotic expression which improves with increasing population size n (Goldberg

and Deb) [20].

$$t^* = \frac{1}{\ln s}[\ln n + \ln(\ln n)] \tag{2.21}$$

The idea of *Elitism* was first introduced by DeJong [24], which forces *genetic algorithms* to retain some number of the best individuals at each generation. Such individuals can be lost if they are not selected to reproduce or if they are destroyed by *crossover* or *mutation.*

Stochastic Remainder "Roulette Wheel" selection, *Stochastic Universal Sampling* , *Binary Tournament* selection are the more popularly used selection procedures considering *time complexity* as well as selection performance. In our study we have incorporated the binary tournament selection along with elitism, and also stochastic based selection operators. Different selection schemes assign different number of copies to better strings in the population, but in all selection schemes the essential idea is that more copies are allocated to the string with higher fitness values. After reproduction the population is enriched with clones of good strings which are subjected to *crossover* and *mutation.*

2.2.2 CROSSOVER

Genetic Algorithms employ some form of *crossover-* an operator that combines portions of the parental chromosomes to produce offsprings [17]. In crossover operator, new and expectedly better strings are created by exchanging information among the parental chromosomes selected by the reproduction operator for the mating pool. In a *Single Point Crossover* operator, generally two strings are picked from the mating pool at random and a crossing site (|) along the string is chosen randomly. Now a uniform random number is generated between 0 and 1, and if the random number $r < p_c$ (probability of crossover), then the operator exchanges the subsequences of the two string on the right side of the crossing site [17]. For example, for two five bit strings

Before crossover :

0 0 | 0 0 0

1 1 | 1 1 1

after crossover :

0 0 1 1 1

1 1 0 0 0

where the crossing site is chosen randomly after the second locus of the strings. Basically, crossover mimics biological recombination between two single chromosome(*haploid*) organisms [19].

In order not to destroy previously found good strings of the mating pool, crossover is usually performed with a probability $p_c < 1$ (a pre-assigned value), because depending upon the crossing site chosen randomly, the crossover may have some detrimental effect along with its beneficial effect. The idea of crossover is to recombine building blocks (schemas) of different parental chromosome in the offsprings. The single point crossover operator has some bias of exchange for the right-most bits. They have a higher probability of getting exchanged than the left-most bits in the string. Single point crossover keeps intact short, low-order schemas as the functional *building blocks* of the strings; so schemas with long defining lengths are likely to be destroyed under this operator [19]. This shortcomings can lead to the preservation of the *hitchhikers* [1]. It has also been observed in *" ROYAL ROAD"* experiments by Mitchell et al. [17]. Single point crossover often treats some *loci* (crossing site) preferentially, as a result the segments exchanged between the two parents always contain the *end-point* of the strings.

To reduce the *positional bias*, and *end point effect*, *two point crossover* is sometimes preferred, in which two positions are chosen at random and the segments between them are exchanged. The two point crossover is less likely to disrupt schemas with large defining lengths and can combine more schemas than single point crossover. In addition, the segments that are exchanged do not necessarily contain the *end points* of the strings. Again, there are

[1] bits that are not part of a desired schema but which, by being close to the string, hitchhike along with the beneficial schemas it produces

schemas that two point crossover can not combine. In order to overcome those difficulties, *multi point crossover* is often used. Example of two point crossover is given below

Before crossover :
0 | 0 0 | 0 0
1 | 1 1 | 1 1
after cross-over :
0 1 1 0 0
1 0 0 1 1

Extension of the above crossover operators is the *parameterized uniform crossover* [17] where an exchange happens at any bit location from either parent with a probability $p_c = 0.5$. Parameterized uniform crossover has no positional bias , any schemas contained at different positions in the parents can be *potentially recombined* in the offsprings [19]. An example of the uniform crossover operator is given below, where the first and the fourth bit positions have been exchanged.

Before crossover :
0 0 0 0 0
1 1 1 1 1
after cross-over :
1 0 0 1 0
0 1 1 0 1

this operator has the maximum search power among all of the above crossover operators, but simultaneously has the minimum survival probability for good bit combinations (schemas) from parents to offsprings. In our present study single point crossover operator is used along with the option for uniform crossover.

2.2.3 MUTATION

The crossover operator is generally regarded as the major instrument of variation and innovation in the population of the *genetic algorithms*, while the *mutation* operator prevents the population from permanent fixation at any particular locus and thus plays more of a

background role [17]. Mutation operator is used sparingly and changes 1 to 0 and vice-versa, with a small pre-assigned probability p_m. The bitwise mutation or *jump mutation* is performed bit by bit by simulating of flipping a coin with probability p_m. The simulation of flipping a coin is done by generating a uniform random number between 0 and 1, and if the random number is smaller than p_m, then the outcome of the flipping is true, otherwise the outcome is false. For any bit if the outcome is true, then the bit is altered, otherwise kept unchanged. The need of mutation is to create a point in the neighborhood of the current point, thereby achieving a local search around the current solution. It also introduces diversity in the population whenever the population tends to become homogeneous due to repitative use of the selection and crossover operators. The inclusion of mutation introduces some probability (Np_m) of turning 0 to 1, where N is the population size.

Another variant of mutation, called *creep mutation* is also used. In creep mutation each variable is examined in the real space, and is either increased or decreased by a small value with a probability of p_{creep} and then reconverted into the binary space.

After new strings are created, they are evaluated by decoding and calculating the objective function values. This completes one cycle of *genetic algorithms* iteration, known as *generation* in *genetic algorithms* terminology, the same process continues until a termination criterion is satisfied. In next few sections some advanced operators used in *genetic algorithms* and some modern developments in this methodology are discussed.

2.2.4 MICRO GA

A *Simple Genetic Algorithms* (SGA) works with a serially implemented binary coded population of size N, with a generation to generation evolution based on reproduction, crossover and mutation. The performance of SGA can be measured in terms of on-line and off-line performance measures (the on-line performance measure is the average performance measure of of all the tested structures over the course of the search; the off-line performance

is the average of all the generation-based best performance structures). For the function optimization problems, where the function to be optimized is well defined and do not change faster than the evolution process itself, SGA can be implemented very easily. SGA therefore can handle *stationary function* optimization problems quite satisfactorily to find the exact optimum. However, for the *non-stationary function* optimization problems, where the functions to be optimized are themselves evolving at a much faster rate, SGA can not find an optimum. Problems of this type are found in many real world situations, such as in aerospace (pursuit and evasion), on-line air-craft trajectory optimization, and the optimal control of a aircraft in wind shear etc. Goldberg has proposed that for serial implementation of binary GAs the optimal population choice is small, which was obtained from optimizing effective real-time schema processing in a given population. He also pointed out that simply by taking a small population size and letting them converge is not very useful and outlined a scheme by which a small population GA can be implemented. Based on this approach, a step by step procedure for the $\mu-GA$ implementation, as proposed by Krishna Kumar, is given below [16].

1. Select a population of size 5 either randomly or 4 randomly and one good string from any previous search.

2. Evaluate fitness and determine the best string. Label it as string 5 and carry it to the next generation (elitist selection strategy). In this way there is guarantee that the information about good schema is not lost.

3. Choose the remaining four strings for reproduction (the best string also competes for a copy in the reproduction) based on a deterministic tournament selection strategy [19]. Since the population is small, the law of averages does not hold good and the selection procedure is purely deterministic. In the tournament selection, the strings are grouped randomly and adjacent pairs are made to compete for the final four (Care should be taken to avoid two copies of the same string mating for the next generation).

4. Apply crossover with $p_c = 1$. This is done to facilitate a high order of schema processing. The mutation rate is kept to zero as it is clear that enough diversity is introduced after every convergence through new population of strings.

5. Check for nominal convergence (reasonable measure based on either genotype convergence or phenotype convergence). If converged go to step 1, else go to step 6.

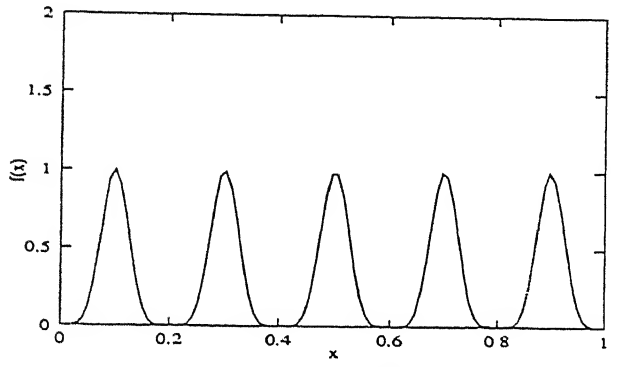
6. Go to step 2.

2.2.5 NICHING AND SHARING

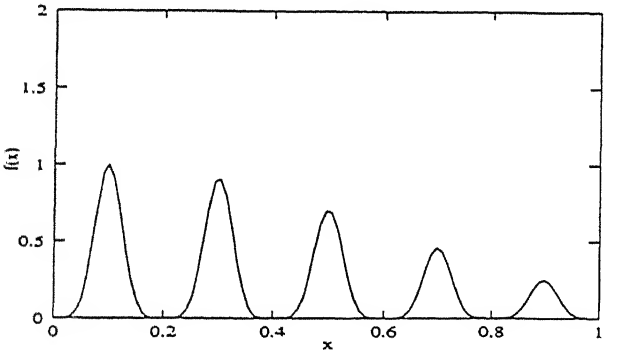
In artificial genetic search, multi-modal functions are optimized by introducing the natural concepts of *niche* and *species* into a population of strings. To achieve better performance on non-stationary functions, *dominance* and *diploidy* have been added [19]. To overcome this limitations in fixed codings, *inversion* and *reordering* operator have been suggested by Goldberg. In dealing with multi-modal functions, simple *genetic algorithms* converge to a single peak [19], even when multiple peaks of equality may exist. Faced with a similar problem, nature forms stable *subpopulations* of organisms surrounding separate *niches* by forcing similar individuals to *share* available resources [17]. In nature, *niche* is defined as an organism's task in the environment and a *species* is a collection of organisms with similar features. The subdivision of environment on the basis of an organism's role reduces inter-species competition for environmental resources, and this reduction in competition helps stable subpopulations to form around different *niches* in the environment [21].

In the optimization of multi-modal functions as shown in Figure 2.4, a *simple genetic algorithms* cannot maintain controlled competition among the competing schemata corresponding to different peaks, and the stochastic error associated with the genetic operators, called *genetic drift* , for finite population. causes the population to converge to one alternative or other [21].

In DeJong's *Crowding* scheme, separate niches are created by replacing existing strings according to their similarity with others in an overlapping population. *Generation Gap* (G) dictates the use of an overlapping model in which only a proportion G of the population is permitted to reproduce in each generation [21].



(a) equal peaks



(b) unequal peaks

Figure 2.4: Sample functions where stable, relatively no competitive subpopulation might be useful (Source: [19, 21]); (a) function with equal peaks for single variable (expected, subpopulation to be equal in size);(b) function with unequal peaks for single variable (expected, subpopulation sizes to decrease with decreasing peak size);

When selecting an individual to die, *crowding factor (CF)* determines the number of individuals that are to be picked at random from the population, and the one which is very much similar to the new individual is chosen for replacement. Here similarity criteria is

defined in terms of the number of matching *alleles* [1]. The new individual then replaces this chosen individual in the population.

Depending upon the Holland's *sharing concept*, Goldberg and Richardson incorporated the *sharing* function by dividing the population in different subpopulations according to the similarity of the individuals in two possible solution space : the decoded parameter space (*Phenotypic Sharing*) and the binary or gene space (*Genotypic Sharing*). A sharing parameter σ_{share} is defined to control the extent of sharing, and a *power-law sharing function, Sh(d)* is defined as a function of *distance-metric (d)* between two individuals

$$Sh(d) = \begin{cases} 1.0 - \left(\frac{d}{\sigma_{share}}\right)^{\alpha} & \text{if d} < \sigma_{share} \\ 0.0 & \text{if d} \geq \sigma_{share} \end{cases} \tag{2.22}$$

Sharing requires lowering of an individual's payoff (fitness) due to presence of its neighbors in the search space. The parameter σ_{share} can be taken as the maximum distance between the strings necessary to form as many niches as there are peaks in the multi-modal solution space.

2.2.6 PHENOTYPIC SHARING

When the proximity of the of the individuals is defined in the decoded parameter i.e. real space, it is called phenotypic sharing. The distance metric ($d_{i,j}$) is defined as the distance between strings in the decoded parameter space. For a p-variable function, the *distance norm* in the p-dimensional space is calculated by considering the Euclidean distance.

$$d_{ij} = \sqrt{\sum_{k=1}^{p} (x_{k,i} - x_{k,j})^2} \tag{2.23}$$

where $x_i = [x_{1,i}, x_{2,i}, ...x_{p,i}]$, $x_j = [x_{1,j}, x_{2,j}, ...x_{p,j}]$ and $x_{1,i}$,$x_{2,i}$,..... ....$x_{p,i}$ are the decoded parameters. If each niche is enclosed in a p-dimensional hypersphere of radius σ_{share} such that each sphere encloses $\frac{1}{q}$th volume of the space, where q is the number of peaks in the solution space. The radius of the hypersphere is calculated as [19, 21]

$$r = \frac{1}{2}\sqrt{\sum_{k=1}^{p} (x_{k,max} - x_{k,min})^2} \tag{2.24}$$

[1] binary bits

and the volume is $V = cr^p$ where c is a constant. Thus σ_{share} can be obtained as

$$c\sigma_{share}^p = \frac{1}{q} cr^p$$

$$\sigma_{share} = \frac{r}{\sqrt[p]{q}}$$

So one will obtain from the above

$$\sigma_{share} = \frac{\sqrt{\sum_{k=1}^{p} (x_{k,max} - x_{k,min})^2}}{2\sqrt[p]{q}} \tag{2.25}$$

2.2.7 GENOTYPIC SHARING

As described by Goldberg and Richardson [21], the *genetic closeness* of the two individuals are taken as the number of different *alleles* in their chromosomes. The distance metric $d_{i,j}$ is defined as the *Hamming distance* between the strings and σ_{share} is the maximum number of different bits allowed between the strings to form separate niches in the population. Comparing the two strings s_i and s_j of string length l, for allowable one bit difference the total number of strings genetically close are $\binom{l}{1}$. For k distinct allowable bits of difference, total $\binom{l}{k}$ strings are possible in the solution space. As the total number of strings possible is 2^l , then for allowable k-bits $\binom{l}{k}/2^l$ strings are away from each other. Thus for all the strings in the population with k bit differences or less is given by $\sum_{i=0}^{k} \binom{l}{i}/2^l$. According to sharing, for q-peaked function, there are q niches in the solution space. Assuming uniform niche placement[1], if k (or σ_{share}) number of bits of differences allowed between the strings to make q-subspaces in the solution space, then [21]

$$\frac{1}{2^l} \sum_{i=0}^{k} \binom{l}{i} = \frac{1}{q} \tag{2.26}$$

2.3 FUNDAMENTALS OF DIFFERENTIAL EVOLUTION

Differential Evolution (DE) is a simple evolution strategy that promises to make fast and robust numerical optimization. The overall structure of DE involves population based

[1]each niche must correspond to an average of $\frac{1}{q}$ of the total solution space

searches. The parallel version of DE maintains two real valued D-dimensional arrays of population number N. The *primary array* holds the current vector population and the *secondary* one accumulates vectors that are selected for the next generation[14]. In each generation N competitions are held to determine the population for the next generation. The *i-th* competition picks the *i-th* population vector, called the *target vector*, and its adversary is known as the *trial vector*. The target vector that competes against the trial vector is also one of the trial vector's parent. The trial vector's other parent is a randomly chosen population vector to which a weighted random difference vector has been added. *Mating* is done between this noisy random vector and the target vector, and is controlled by a *non-uniform crossover operation* that determines which trial vector parameters are inherited from which parent. Three factors those control the evolution under DE are *population size(N), the weight applied to the random differential(F), and the constant that mediates the crossover operator(CR)*. The basic mechanism of DE has been explained in Figure 2.5.

During *initialization*, the parameter limits are set and each parameter in every primary array vector is initialized with a uniformly distributed random value within the allowed range. To determine and preserve the *cost* (function value) of the initial population, each primary array vector is evaluated. Unless not restricted, DE can search beyond the specified limits.

In order to perform *mutation*, DE uses an appropriately scaled perturbation of the population itself. Every pair of randomly chosen vectors $(\bar{X}_i, \bar{X}_j)$ gives rise to a *vector differential* $(\bar{X}_i - \bar{X}_j)$, and their *weighted difference* is used to mutate another vector $\bar{X}_k$, such that

$$\bar{X}'_k = \bar{X}_k + F(\bar{X}_i - \bar{X}_j) \tag{2.27}$$

The scaling factor F is a constant defined in the range $0 < F \leq 1.2$, where the upper limit is determined empirically. The optional value of F lies in the range of 0.4 to 1.0 [14].

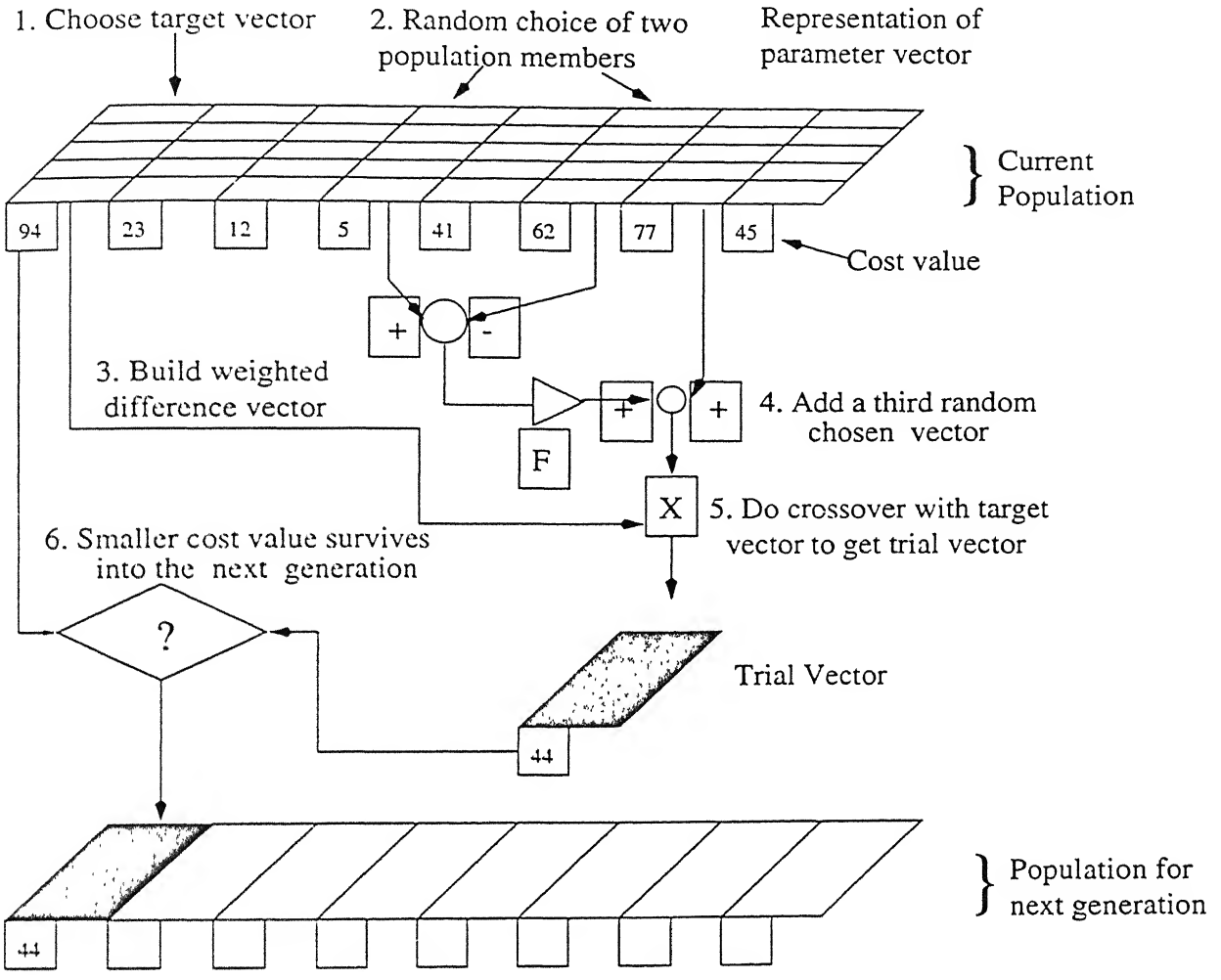


Figure 2.5: The basic mechanism of Differential Evolution (Source:[14]).

By mutating vectors, with population based differential, DE ensures that the solution space will be efficiently searched in each dimension.

In every generation, each primary array vector $\bar{X}_i$ is targeted for *recombination* with a vector $\bar{X}'_k$ to produce a trial vector $\bar{X}_t$. The trial vector (a noisy random vector) is the *child* of two *parents* and it must compete with the target vector. In order to determine the parental contribution for the trial vector parameters a series of *D-1 binomial experiments* [14] are done, whose outcome is either success or failure and its probability is determined by the

crossover constant (CR), where $0 \leq CR \leq 1$. Starting at a randomly selected parameter, the source of each trial vector is determined by comparing the value of CR to a uniformly generated random number within the interval (0 , 1). If the random number is greater than CR, the trial vector gets its parameters from the target $\bar{X}_i$, otherwise the parameters come from $\bar{X}'_k$ (noisy random vector). To ensure $\bar{X}_t$ differs from $\bar{X}_i$ by at least one parameter, the final trial vector parameter always comes from the noisy random vector, and therefore only D-1 binomial experiments are performed in a D variable problem.

For *selection*, the cost of each trial vector is compared with its parent target vector. The vector with more optimal cost is allowed to the secondary array. After each primary array vector has been a target for *mutation*, *recombination*, and *selection*, the array pointers are swapped so that roles of the two arrays are reversed. Thus, vectors in what was the secondary array becomes the target for *transformation*, while the former primary vector competes with the winners of the next generation.

2.4 FUNDAMENTALS OF SIMULATED ANNEALING

The *Simulated Annealing* (SA) optimization algorithm is analogous to the physical process by which a material changes its state while minimizing its energy. A slow cooling brings the material to a highly ordered, crystalline state of lowest energy, while a rapid cooling yields defects inside the material.

Considering $\bar{X}$ be a vector in R^n where $(x_1, x_2.....x_n)$ are its components and $f(X)$ be the function to be optimized where $a_1 \leq x_1 \leq b_1$,.. $a_n \leq x_n \leq b_n$ be its n variables, each ranging in a finite, continuous interval. Although the function $f(X)$ does not need to be continuous but it must be bounded. The SA algorithm for optimizing the function is shown schematically in Figure 2.6. It proceeds iteratively, starting from a given point X_0, and then generates a succession of points $X_0, X_1, X_2, ...X_i, ...$ leading towards the global optimum of the cost function. New candidate points are generated around the current point

X_i by applying random moves along each coordinate direction. The new coordinate values are uniformly distributed in the intervals centered around the corresponding coordinate X_i. Half the size of these intervals along each coordinate is recorded in the step vector V_m. If the point falls outside the definition domain of f, a new point is generated randomly until a point belonging to the domain is found. A candidate point X is accepted or rejected according to the *Metropolis Criteria*, described below [22].

If $\Delta f \leq 0$, then accept the new point $X_{i+1} = X'_i$ else accept the new point with probability

$$p(\Delta f) = Exp(\frac{-\Delta f}{T}) \tag{2.28}$$

where, $\Delta f = f(X') - f(X_i)$ and T is *temperature* like parameter. At a fixed value of T the successive points $X_0, X_1, X_2, ...X_i, ...$ is not always downhill, except $T = 0$. When value of T is large compared with mean value of $|f(X_h) - f(X_k)|$, where X_h and X_k are points randomly chosen inside the definition domain of f, then almost all the new points are accepted. SA starts with a high temperature value T_0 and a sequence of points are generated until an *equilibrium* is reached at that temperature. During this phase, the step vector V_m is periodically adjusted to better follow the function behavior. The best point reached so far is recorded as X_{opt}. After thermal equilibrium is reached at T_o, the temperature is reduced and a new sequence of moves are made starting from X_{opt}, until thermal equilibrium is reached again [1]. The process is stopped at a temperature low enough that no more improvements are expected, according to a stopping criteria.

SA permits occasional uphill moves under the control of a temperature parameter. At a higher temperature only the gross behavior of the cost function is relevant to the search. As the temperature decreases, finer details can be developed to get a good final point. While the optimality of the final point cannot be granted, the method is capable to proceed towards better minima even in the presence of many local minima.

[1] the number of function evaluations until next temperature reduction is the product of total coordinate directions, number of cycles in one iteration and number of iterations allowed

For *step adjustment*, according to *Metropolis criteria* [23], a higher number of accepted moves with respect to rejected ones means the function is explored with too small steps, and on the contrary, a higher number of rejected moves means that new trial points are generated too far from the current point. A [1:1] rate between accepted and rejected moves means that the algorithm is following the *function behavior* well. A step vector V_m records the maximum incremental possible along each direction and is adjusted every N_s-th move to maintain the ratio.

2.5 THE METHOD OF OBJECTIVE WEIGHING FOR MORE THAN ONE OBJECTIVE FUNCTIONS

For Multi-objective optimization problems the methodology of problem formulation is given below

Minimize or Maximize $f_i(X)$ for i = 1,2,....,N

Subject to

$$\left.\begin{array}{l} g_j(X) \leq 0.0 \quad \mathrm{j} = 1, 2, ..J \\ h_l(X) = 0.0 \quad \mathrm{l} = 1, 2...L \end{array}\right\} \tag{2.29}$$

the parameter X is a p dimensional vector, having p design or decision variables.

In the Classical approach methodology the easiest way of handling such kind of problems is *The Method of Objective Weighing*, where the over-all objective function

$$Z = \Sigma_{i=1}^{N} w_i f_i(X)$$

where $x \in X$ and X represents the feasible search region. In the above formualtion, $0 \leq w_i \leq 1.0$, and $\Sigma_{i=1}^{N} w_i = 1$.

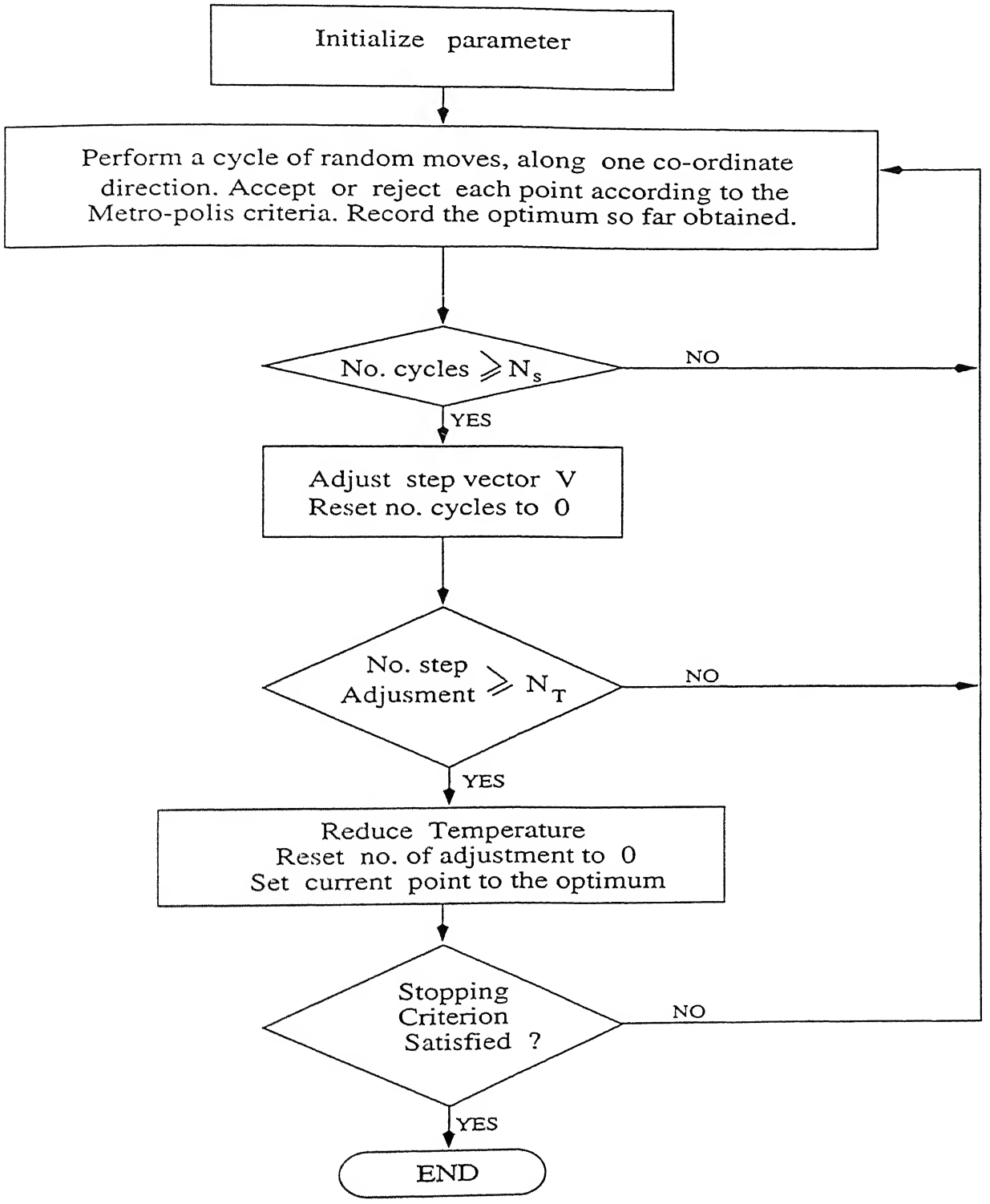


Figure 2.6: The flow chart for the Simulated Annealing Algorithm for Multi modal Function optimization (Source: [22]).

Chapter 3

Mold Design Problem

3.1 Basic Aspects of Problem Formulation

The liquid steel begins to solidify as it enters the mold and forms the solidified strand shell. When the strand leaves the mold, the thickness varies between 10 to 30 mm, depending upon the casting speed [1].

According to the product requirement of steel plant, a wide variety of cross sectional shapes and sizes can be continuously cast. The mold determines the cross section of a strand. Different types of molds are required to cast different types of strand, such as rectangular, square, round, hollow or polygon in cross section. Molds are generally classified into three categories [1]

- Block molds.
- Tubular molds.
- Plate molds.

Shaping wall of the molds are made of rolled or forged plates of copper alloys. These are bolted with a steel box which acts as cooling water jacket. The water jackets are supported by a steel frame on to which the gear boxes for the narrow face and the clamping mechanisms are mounted.

The steel water boxes perform a dual function [10]

(i) They provide a stable support for the copper liners.

(ii) Ensure even distribution of the mold cooling water over cooling surfaces.

Mold lengths vary between 800 and 1000 mm depending on the production requirements [1]. The length is usually determined as a function of the casting speed. From heat transfer measurements, it was suggested that high casting speed requires a longer mold [8].

Mold oscillation is produced by means of cam-rollers. As a result, sinusoidal motion of the mold is produced, which is shown in Figure 3.1.

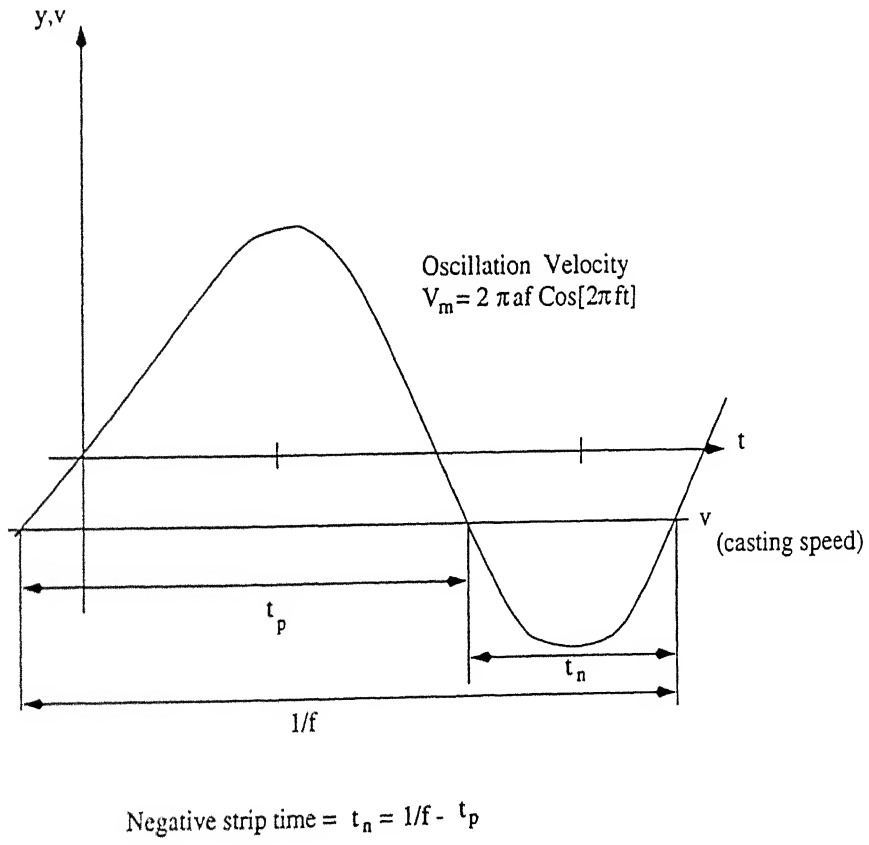


Figure 3.1: The essential features of mold oscillation (Source: [1]).

Movement of the mold prevents sticking of the solidified shell to the mold wall. Other-

wise, tensile stresses produced in the solidified shell leads to surface cracking of the strand. Parameters involved in this reciprocal motion are the frequency (F) and the stroke length (S). Surface quality of the product largely depends on the correct choice of these parameters. During downward movement of the mold (*negative strip*), axial pressure is applied to the solidified shell, which prevents transverse cracking of the strand. Oscillation marks are produced from the interaction between the casting slag and the mold movement. The magnitude of the negative strip time was calculated by [10, 45, 46] as

$$T_N = \frac{1}{\pi F} cos^{-1}\left(-\frac{V_c}{\pi F S}\right) \tag{3.1}$$

where, T_N is the negative strip time in (sec), F is the frequency of the mold in (cycles/sec) and V_c is the casting speed in (mm/sec). Short strokes or small amplitudes at high frequency have been successful in minimizing the oscillation marks [1]. In order to obtain optimum negative strip time T_N of 0.2 to 0.3 seconds under fluctuating casting speed, mold movement and casting speed are synchronized [1, 45].

Mold fluxes are synthetic slags used during the continuous casting of steel. Fluxes are continually fed on the liquid pool surface during casting. As a result, it melts and liquid slag is formed. The liquid slag flows down between the mold wall and the solidified shell. Figure 3.2 schematically shows the general disposition of a flux in the continuous casting mold [26, 27].

The flux above the surface of the molten steel consists of three different layers [44].

(i) An unmelted, dark, unreacted powder layer on top.

(ii) A heterogeneous sintered layer in the middle.

(iii) A molten flux layer over the liquid steel.

A rim is formed by flux in contact with the water-cooled copper mold walls at the meniscus level, which has a considerable influence on heat transfer process at the meniscus [28].

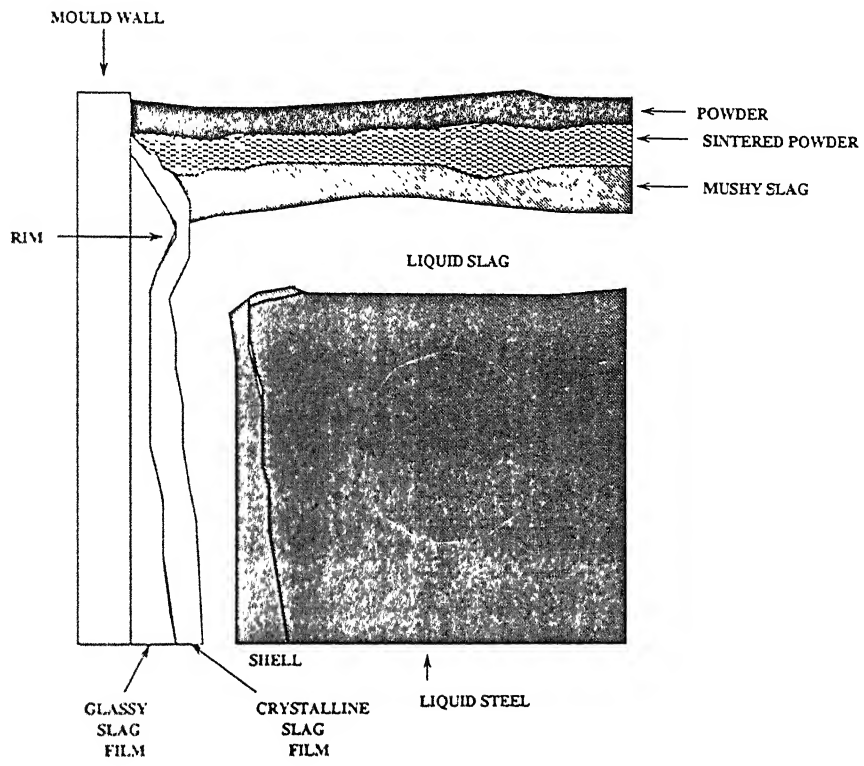


Figure 3.2: The schematic representation of flux behavior in a continuous casting mold (Source: [26])

Below the meniscus, flux layer consists of a solid film directly in contact with mold walls and a liquid film in contact with the strand surface [26, 28]. Mold fluxes are used for several purposes [44].

(i) provide thermal insulation.

(ii) protect the liquid steel against re-oxidation.

(iii) absorb the non-metallic inclusions.

(iv) lubricate the strand and reducing strand friction.

(v) control the heat transfer between strand and mold.

Ogibayashi et al. [34] investigated the melting behavior of *prefused* type, granulated powders for different viscosity range and *vitrification ratio* between 25 and 90%. The vitrification ratio was defined as the percentage of molten flux formed when 25 gm of powder were unidirectionally heated at $1400^{o}C$ for seven minutes in an alumina crucible [44]. After heating, the crucible was cooled and cut through the center and the height or % of liquid flux is taken as an index of melting rate. The percentage of liquid flux formed was termed as *vitrification rate* by the researchers at Nippon Steel [31, 35]. When the vitrification rate exceeds 80%, the flux tends to form a flux rim which causes partial solidification of liquid steel surface in a slab caster [44]. When vitrification rate is too low, the longitudinal cracks are formed. In order to prevent such defects, Ogibayashi et al. [34] established an optimum flux pool depth of approximately 10 mm corresponding to an optimum vitrification rate of about 60%.

Once the flux has been added to the mold, two major phenomena occur [44]

(i) mold powder is heated in contact with the liquid metal, melts and feeds the surface with a liquid phase.

(ii) the liquid phase spreads over the steel surface and infiltrates into the gap between strand and mold during oscillation cycle.

Though the infiltrated flux solidifies in contact with water-cooled copper mold wall and forms a solid layer, a molten flux layer remains which ensures lubrication of the strand-mold interface [35]. The molten flux pool is formed above the meniscus of steel which acts as a reservoir and ideally provides a continuous supply of liquid flux to the mold-strand gap. It is important that this flux-pool has sufficient depth to feed the gap adequately all the times [36]. Mold powder consumption rate is related to the thickness of the liquid film, as well as to the depth of oscillation marks. The powder consumption rate decreases with the increasing casting speed and a reduction in oscillation frequency and negative strip time causes the increase in powder consumption rate [32, 36].

Nakano et.al [35] also showed that the agitation on the liquid steel surface accelerated the melting of mold fluxes and increase the thickness of flux pool. A flux pool thickness in the 10 to 15 mm range has been recommended as an optimum value [13, 29, 33, 34, 38]. For higher casting speed Mills recommended a minimum thickness of 20 mm. Bommaraju [30] recommended a minimum thickness of 6 to 12 mm, which is in the same range reported by Tsai and Mastervich [37]. Nakano et al. [35] suggested a minimum flux pool depth (Y_p) is given by

$$y_p = S sin\left(\frac{\pi N}{2}\right) - \frac{500 N V_c}{F} + \delta \tag{3.2}$$

where S is stroke length of mold oscillation in (mm), F is the frequency of oscillation in (cycles/min), V_c is the casting speed in (m/min), δ is the molten surface oscillation in (mm) and N is the negative strip ratio in (sec), which is given by

$$N = 1 - \left(\frac{2}{\pi}\right) sin^{-1}\left(\frac{1000 V_c}{\pi S F}\right) \tag{3.3}$$

Koyama et al. [31] established the following empirical equation relating the average flux pool thickness with vitrification rate, mold dimensions, casting speed, and powder consumption rate.

$$d = 0.02\left(\frac{S_R}{abwV}\right) \tag{3.4}$$

where, d is the flux pool depth in (mm), S_R is the vitrification ratio in (%), a and b are the transverse mold dimensions in (m), V is the casting speed in (m/min), and w is the consumption rate in (kg/t).

Heat transfer in the mold is one of the most important phenomena taking place during the continuous casting of steel. In order to get slabs with good surface quality, proper heat extraction is necessary. If the rate of heat removal is excessive and uneven, thermally induced stresses are generated in the solidified shell. As a result, longitudinal cracks may occur. Insufficient heat removal can lead to a thin solidified shell prone to bulging or breakouts [39, 40, 41].

From the liquid solid-steel interface heat is transfered to the mold by a sequence of steps consisting of [44].

(i) Convection in the liquid steel pool.

(ii) Conduction through the solidified shell.

(iii) Heat transfer across the flux layer infiltrating the strand-mold gap.

(iv) Heat transfer across any air gap separating the strand and the mold.

(v) Conduction through the mold walls.

(vi) Convection at the mold-cooling water interface.

The largest resistance to heat extraction is encountered between the solid shell and mold wall [39, 40, 43]. Therefore the mold heat transfer is controlled by those factors which determine the thickness of the gap separating the solidified shell and the copper mold wall and the properties of the flux infiltrating the gap.

The basic aim of the continuous casting mold design problem cosidered in this study is to maximize the speed of casting while keeping the various operational variables within some required range. This basic goal can be achieved by formulating the continuous casting mold design problem as an optimization problem. In the following three different formulations of the mold design are presented.

3.1.1 FIRST FORMULATION

Considering the empirical equations given by Nakano et.al [35] and Koyama et.al [31], the first objective function is formulated here as follows.

Maximize,

$$V = \frac{0.00157 s^2 f sin(\pi N)}{Y_p - \delta + (\frac{10.0 S_R N}{abdwf})} \tag{3.5}$$

where, V is the casting velocity. We formulated the constraints from a physical consideration of the mold behavior. The constraints being considered are as follows:

$$g(1) = (\frac{Y_p}{d}) - 0.99 \geq 0.0 \tag{3.6}$$

$$g(2) = 1.001 - (\frac{Y_p}{d}) \geq 0.0 \tag{3.7}$$

where, d is the flux pool depth in (mm), Y_p is the minimum liquid pool depth in (mm), s is the stroke length in (mm), f is the frequency of oscillation in (cpm) , δ is the molten steel

surface oscillation in (mm), N is the negative strip time in (sec), S_R is the vitrification ratio in (%) and w is the consumption ratio in (Kg/ton). The parameter (ab) is the size of the billet being casted (in sq.m.)

However, in this formualtion the basic mechanisms of heat transfer can not be considered. This is not good enough to study the solidification process takes place in the primary cooling region of a continuous caster. Hence, in order to incorporate the heat extraction mechanisms into the problem description the second formulation is done. The formulation is presented in the next section.

3.1.2 SECOND FORMULATION

The configuration of the mold region is shown schematically in Figure 3.3. Heat balance has been done for this configuration following the approach of Geiger and Poirier [49], in order to get the governing equation for the casting speed in terms of various operational parameters as shown below. For a considerable ranges of solidified shell formation the casting speed is optimized in this study. Furthermore, for a very thin solidified shell thickness the strand may bulge out and a very high solidified shell will reduce the casting speed making the caster an inefficient one.

While in the mold, the conduction of heat in the thin shell of solid is much greater in the x-direction than in the y-direction. Therefore the conduction of heat in the withdrawal direction may be ignored. As the metal passes through the mold, it can be assumed that a constant heat transfer coefficient applies to account for the mold-casting interface resistance, as heat is removed by the water-cooled mold maintained at T_o.

The interfacial resistance predominates over the resistance offered by the solidifying metal [49].The temperature gradients within the mold and the casting are negligible. Thus the total amount of heat Q that transfers the mold-casting interface in time t is

$$Q = hA(T_s - T_o)t \tag{3.8}$$

If the temperature gradients are negligible, then $T_s \cong T_m$, and only latent heat is removed

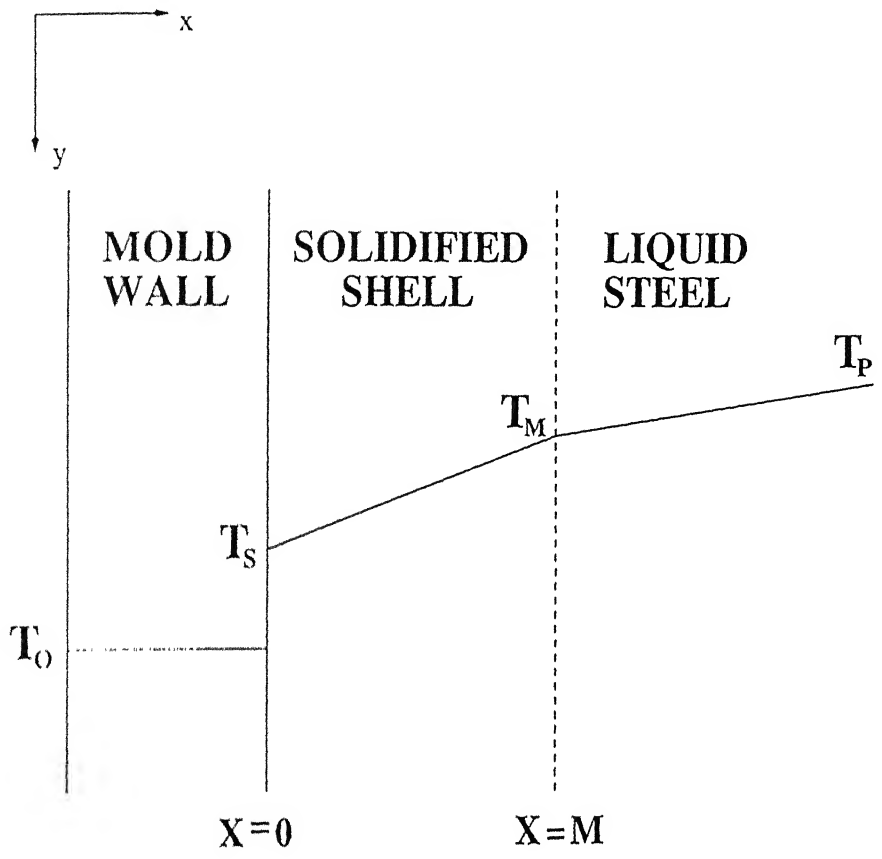


Figure 3.3: Configuration of the mold used in the second objective function formulation

from the casting during solidification. As $Q = \rho' v H_f$, then we can obtain

$$\frac{v}{A} = \frac{h(T_M - T_o)}{\rho' H_f} t \tag{3.9}$$

where ρ' is the density of the solidifying steel, v is the volume of steel solidified, A is the cross-sectional area and H_f is the latent heat of fusion and h is the heat-transfer coefficient. As the shape has no effect, Equation (3.9) can be applied for the uni-directional solidification in the form

$$M = \frac{h(T_M - T_o)}{\rho' H_f} t \tag{3.10}$$

where M is the thickness solidified. Now, considering the case $T_s \neq T_M$, and heat leaves the casting by convection at the surface to a water-cooled mold maintained at T_o, we can

calculate the heat flux at the casting-mold interface as

$$q|_{x=0} = k' \frac{T_M - T_s}{M} \tag{3.11}$$

$$q|_{x=0} = h(T_s - T_o) \tag{3.12}$$

Now, combining the Equations (3.11) and (3.12) and eliminating the surface temperature T_s, and assuming the linear temperature profile, we obtain the flux at the solid-liquid interface as

$$q|_{x=0} = q|_{x=M} = \frac{T_M - T_o}{\frac{1}{h} + \frac{M}{k'}} \tag{3.13}$$

Also, at $x = M$, the latent heat is evolved such that,

$$q|_{x=M} = \rho' H_f \frac{dM}{dt} \tag{3.14}$$

By combining the two equations and integrating with $M = 0$ at $t = 0$ and $M = M$ at $t = t$ we obtain [49],

$$M = \frac{h(T_M - T_o)}{\rho' H_f} t - \frac{h}{2k'} M^2 \tag{3.15}$$

According to Adams [49], a more refined analysis yields

$$M = \frac{h(T_M - T_o)}{\rho' H_f a} t - \frac{h}{2k'} M^2 \tag{3.16}$$

where the parameter a can be defined as

$$a = \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{C'_P(T_M - T_o)}{3H'_f}} \tag{3.17}$$

Now, defining t as the distance down the mold wall L, divided by the casting velocity U, we obtain the second objective function

$$U = \frac{2k'h'(T_M - T_o)L}{\rho' H'_f a(2k'M + M^2 h)} \tag{3.18}$$

where, M is the solidified shell thickness , L is the mold length , T_o and T_M are the temperatures shown in Figure 3.3 , h is the heat transfer co-efficient at the mold-solid

interface , k' is the thermal conductivity of the solidifying metal and ρ' is the density of the solidifying metal .

Assuming that the liquid temperature is uniform with mold length at T_p, and at the solid-liquid interface, latent heat plus liquid superheat is conducted towards the mold wall through the solid skin, we can define the effective latent heat of fusion H_f' as,

$$H_f' = H_f + C_{PL}(T_P - T_M) \tag{3.19}$$

where, H_f is the latent heat of fusion, T_P is the pouring temperature of the liquid metal, C_{PL} is the specific heat.

The Objective function used here :

$$CASTING\ VELOCITY = w_1 U + w_2 V \tag{3.20}$$

where, U and V varies within a very small range. so the another constrained used here is

$$\delta_1 \leq |(U - V)| \leq \delta_2 \tag{3.21}$$

where, δ_1 and δ_2 are varying within a very small range.

The simplified formulation, as presented here provides a reasonable approximation of the mold design problem [3]. However, for a better description of the mold problem we have to consider a number of additional factors [3]. The additional factors, considered in the formulation, are presented in the next section.

3.1.3 THIRD FORMULATION

Following Brimacombe and Samarasekera [51] a more reliable and precise mold configuration is schematically shown in Figure 3.4.

The mushy zone that forms in a continuous caster retains a portion of the latent heat affecting the the temperature profile in the mold region. Among other factors cooling water velocity, thermal conductivity of the mold-powder that fills the gap between the innerwall of the mold and the outer surface of the casting have a crucial role in the mechanism of heat transfer in continuous casting [3]. In order to make a precise approximation, we have formulated the casting speed in the following fashion.

$$U = \frac{2k'(\tilde{h} + \delta_1)(T_M - T_o)L}{\rho' H_f' a[2k'M + M^2(\tilde{h} + \delta_1)]} \tag{3.22}$$

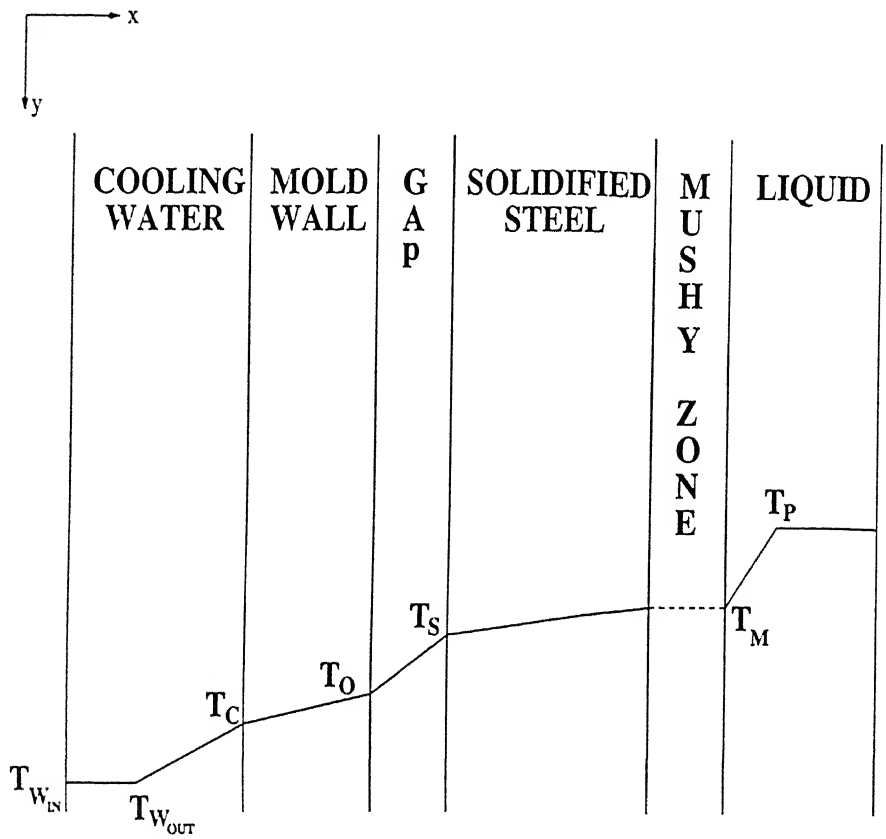


Figure 3.4: The configuration of the mold the third objective function formulation

where $\tilde{h}$ is the modified heat transfer coefficient for the gap region, δ_1 is the heat transfer coefficient correction factor. The available latent heat of fusion was adjusted through a mushy zone-correction factor ε,

$$H_f^{'} = (1-\varepsilon)[H_f + C_{PL}(T_P - T_M)] \tag{3.23}$$

Following the approach adopted by Geiger and Poirier [49], we can consider an imaginary reference between the mold and the casting, which is described in the Figure 3.4 as T_s, where T_s is constant. The contact resistance is then approximated on both sides of the mold-metal interface as

$$\tilde{h} = \frac{h_M + h_C}{2} \tag{3.24}$$

where,

$$h_M = h(1 + \sqrt{\frac{k\rho C_P}{k'\rho' C'_P}}) \tag{3.25}$$

and

$$h_C = h(1 + \sqrt{\frac{k'\rho' C'_P}{k\rho C_P}}) \tag{3.26}$$

where, k is the thermal conductivity, ρ is the density, C_P is the specific heat respectively. The primed quantities are on the steel side while the rests are on the copper mold side. As the value of $\bar{h}$ is only approximate, thus a correction factor δ_1 was included. The parameter 'a' is defined in the following way

$$a = \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{C'_P(T_M - T_o)}{3[(1-\varepsilon)H'_f]}} \tag{3.27}$$

Using these expressions the objective function here is formulated as

$$CASTING\ VELOCITY = w_1 U + w_2 V \tag{3.28}$$

where U is as given in Equation 3.22 and V is as given in Equation 3.5.

3.1.4 THE CONSTRAINTS FOR THE THIRD FORMULATION

The first constraint formulated here is

$$\delta_1 \leq \mid (U - V) \mid \leq \delta_2 \tag{3.29}$$

where, δ_1 and δ_2 are varying within a very small range.

By equating the average mold heat-flux expression (provided by Brimacombe [51]) with heat-flux at the mold-metal interface, the first physical constraint is formulated as [3, 2]

$$2675.2 - 334.4\sqrt{\frac{L}{U}} - (\bar{h} + \delta_1)(T_S - T_o) \tag{3.30}$$

where the time was scaled as $\frac{L}{U}$ and the original value of the constants is converted to SI unit. Assuming the mold-gap thickness (Δg) filled with mold powder of thermal conductivity k_m, we calculate the second constraint as [3]

$$\frac{k_m + \delta_2}{\Delta g} - (\bar{h} + \delta_1) = 0 \tag{3.31}$$

Due to significant uncertainty about k_m a correction factor δ_2 is added to it.

The third equality constraint was formulated by conducting heat-balance at the cooling water side [3]

$$h_w k(T_o - T_{Wout}) - (\rho_w C_{PW} \phi (T_{Wout} - T_{Win}) - \beta)(k + h_w \Delta t_{mold}) = 0 \tag{3.32}$$

Heat transfer between the mold wall and the cooling water takes place by forced convection which has been accounted for by the help of a heat transfer coefficient (h_w) determined using the following correlation [47, 48].

$$h_w = 0.023 \frac{k_w}{D_H} \left(\frac{\rho_w \phi D_H}{\mu_w}\right)^{0.8} \left(\frac{C_{PW} \mu_w}{k_w}\right)^{0.4} \tag{3.33}$$

where, the various temperature terms are shown in the Figure 3.4. ϕ denotes the cooling water velocity (m/sec), Δt_{mold} is the thickness of the mold, β denotes the heat loss adjustment factor, μ is the viscosity, D_H is the hydraulic diameter, and parameters on the cooling waterside are denoted with a subscript w.

In the final formulation the equality constraints are combined together to form

$$2675.2 - 334.4\sqrt{\frac{L}{U}} - \frac{(k_m + \delta_2)}{\Delta g h_w k} [h_w k(T_S - T_{Wout}) - (\rho_w C_{PW} \phi (T_{Wout} - T_{Win}) - \beta)$$

$$(k + h_w \Delta t_{mold})] = 0 \tag{3.34}$$

A total of eighteen important operational parameters including the correction factors have been optimized by using different optimization algorithms, details of which are described in the subsequent chapter. The values of the constants along with references and parameter ranges are given in Tables (A.1-A.4) Appendix A.

Chapter 4

Computational Aspects

In order to get the optimum values of the operational parameters for the multi-variable objective functions used in this study, several optimization approaches have been carried out. These approaches are briefly described below.

4.1 Optimization using Steepest Descent Technique

Penalty function method coupled with a *steepest descent algorithm* was applied first to get the global optimum values. The objective function given in Equation(3.22) has been tried out by this approach first. The equality constraint considered in the formulation given in Equation(3.34), has been transformed into two inequality constraints varying within a very short acceptable range. Thus, the equality constraint $h(X) = 0$, gives rise two inequality constraints

$$g_1(X) = h(X) - \delta_1 \geq 0.0 \tag{4.1}$$

$$g_2(X) = \delta_2 - h(X) \geq 0.0 \tag{4.2}$$

where, δ_1 and δ_2 are varying within a small range and $\delta_2 > \delta_1$. Other inequality constraints have been formulated considering the upper and lower bound of the design variables. In order to obtain same contribution for all the inequality constraints in the penalty term, and to make the search more efficient in the hyper-dimensional search space, *the normaliza-*

tion procedure has been adopted. So, the formulation of the penalty term in Equation(2.2), $G_j[g_j(X)]$ is given by the following equation

$$G_j[g_j(X)] = \frac{g_j(X)}{g_{j,max}(X)} - 1.0 \geq 0.0 \tag{4.3}$$

In this study the *exterior penalty parameter approach* had been adopted to handle the inequality constraints, while the optimization was carried out by using the *Steepest Descent* search procedure.

The multi-modality and differentiability of the objective function used here can be determined from Figures 4.1-4.4. The three dimensional contour plots are drawn by varying two parameters within the specified ranges along x and y axes and plotting the corresponding objective function value, (i.e. the casting speed), along z axis. For all the plots, dimension of the cast billet was taken as 150 sq.mm. The values for the flux pool depth (Y_p), liquid pool depth (d), stroke length of the mold (s), molten steel surface oscillation (δ), vitrification ratio (S_R), mold-flux consumption rate (w) and temperature at the inner surface of the mold (T_o) were kept constant at 15mm, 15mm, 20mm, 10mm, 65%, 0.9 kg/ton, $450K$, respectively, for all the figures.

From the figures, it is evident that the objective function behaves unimodally with the variation of the solidified shell thickness (m), frequency of mold oscillation (f), and negative strip time (N). However, there is a highly non-linear interaction and multi-modality of the objective function in the hyper-dimensional search space with respect to the different correction factors such as δ_1, ε etc.

It is rather well known that the gradient based traditional methods are very much sensitive to proper initial guess vector and also dependent upon the upgradation technique of the penalty parameters. It is obvious from the figures which are describing the gross function behavior, that the *steepest descent method* should be unable to converge properly at the global optimum satisfying all the constraints. Thus, during the actual calculations it was not possible for this search procedure to optimize all the eighteen parameters simultaneously for the multi-variable objective functions used in this study. Therefore, the objective function given in Equation (3.28) could not be used with the *steepest descent* algorithm, a simplified problem with a total of eight variable described in Equation (3.5) was used instead.

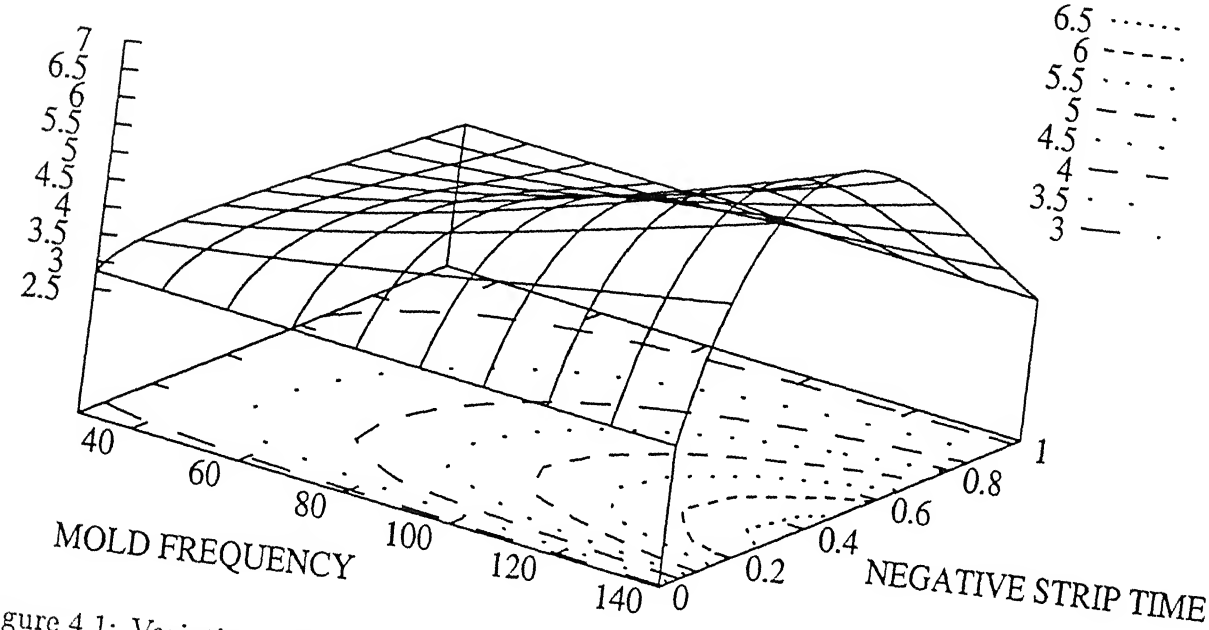


Figure 4.1: Variation of the objective function value i.e. casting speed (m/min) with respect to mold frequency F and negative strip time N

4.2 Optimization using Genetic Algorithms

In this study *genetic algorithms* was successfully applied to all the three objective functions described in Chapter 3. While operating with *genetic algorithms*, the task of optimization has become more easy compared to the traditional methods. This novel evolutionary search methodology has been implemented by linear mapping of the decision variables into a binary search space. Because of this binary encoding the search space has been discretized within the definition domain (X). In *genetic algorithms* the variables have been encoded in binary substrings of length l_i according to the precision required. A *FORTRAN* 77 code [18] available in public domain is mainly applied for finding the global optimum for this particular case. The necessary changes have been made in the code to implement some special features such as splicing, stochastic remainder *"R-W"* selection etc. For reproduction, binary tournament selection operator has been found to perform satisfactorily where one *offspring* has been selected for the mating pool from any two random parental strings in

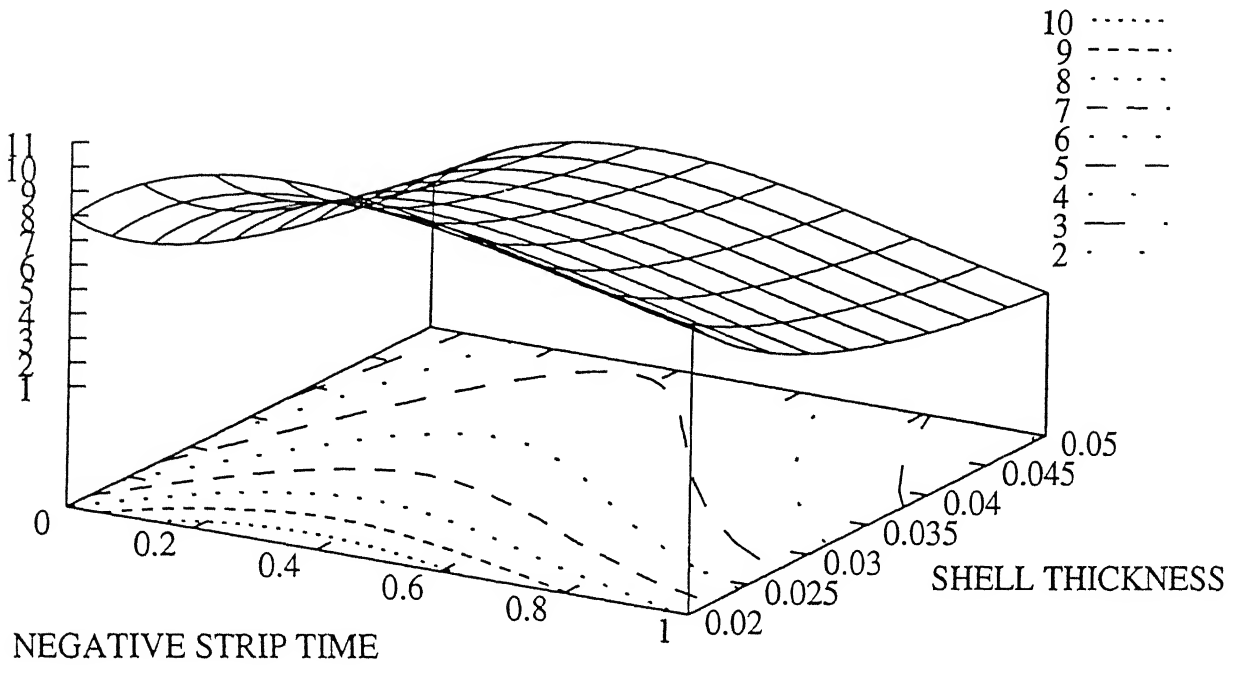


Figure 4.2: Variation of the objective function value i.e. casting speed (m/min) with respect to negative strip time N and solidified shell thickness M

the population. The single point crossover with crossover probability p_c (preassigned value between 0.7 and 0.9) has been found to do well rather than uniform crossover. Jump mutation and creep mutation operators have been employed with low mutation probabilities p_m and p_{creep}, usually taken in the range of 0.001 to 0.01. To handle the constraints, the penalized ϕ-function has been considered as the fitness function during the genetic operations. The penalty parameter R is adjusted properly based upon the value of penalty terms. The number of inequality constraints which has been considered is also less, because in GA the variable bounds are automatically kept fixed while employing the linear mapping scheme. The objective function has been penalized heavily while the constraints are violated.

In this problem there seems to be pronounced tendency of the objective function, i.e. the casting speed. to go into the negative regime, which has no physical significance. In order to overcome this problem, we have assigned a very low fitness value { $O(10^{-9})$} of such individuals. while performing the GA calculations. We have also tried out to find the convergence using μ-GA, with the options for elitist selection and uniform crossover. But

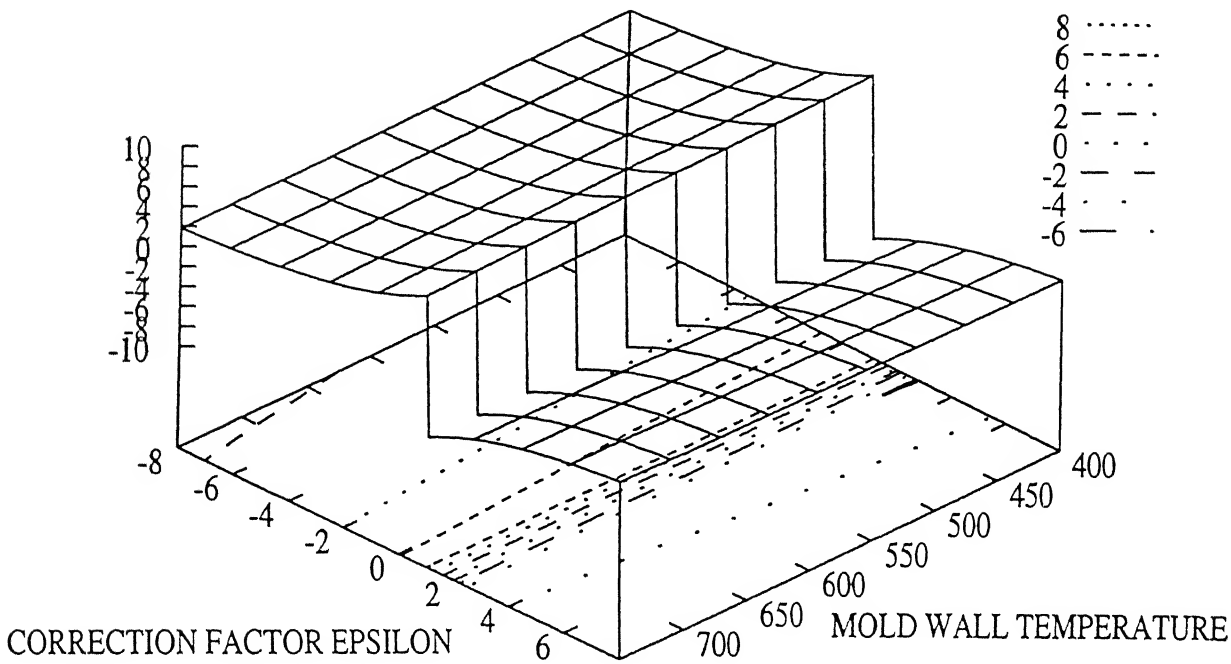


Figure 4.3: Variation of the objective function value i.e. casting speed (m/min) with respect to mold wall temperature T_0 and correction factor ε

the performance of *SGA* appears to be much better than *micro-GA*, where the domain for this stationary objective function is precisely defined.

To optimize all the eighteen parameters, *genetic algorithms* required a population size (N) of 450 individuals and was able to converge after several thousands of generations making the whole process highly computation intensive.

4.3 Optimization using Differential Evolution

In this study the simpler real valued approach of *differential evolution* (DE) has also been applied to find the global optimum. *Differential evolution* has a tendency to search beyond the search-space which has been restricted by fixing the upper and lower bounds for different parameters, by perturbing the population vector generations after generations. Among different strategies of crossover used in DE, the binary crossover strategy has been found to perform satisfactorily. A modular division has been employed for selecting the

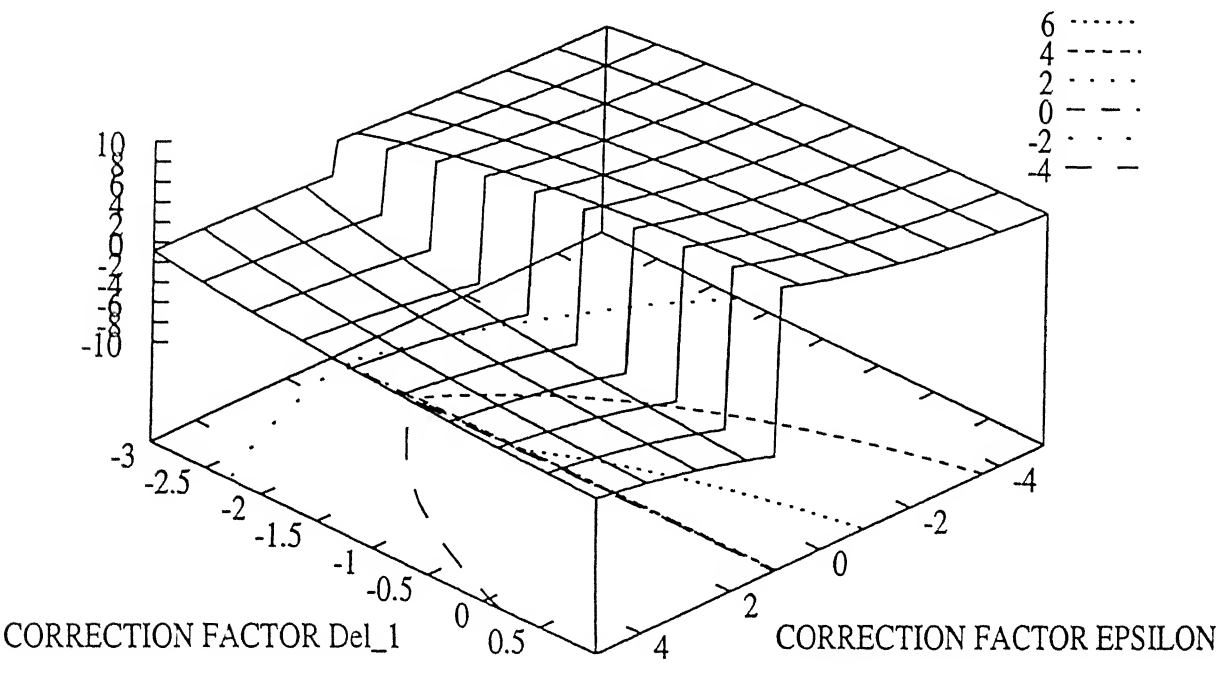


Figure 4.4: Variation of the objective function value i.e. casting speed (m/min) with respect to correction factor ε and correction factor δ_1

parameter value to be taken directly from target to trial vector for every design vector in the population. After thousands of generations DE also has reached the near optimal solution for all the eighteen parameters in the multi-objective function formulation described by Equation(3.28).

4.4 OPTIMIZATION USING SIMULATED ANNEALING

Simulated Annealing, one of the adaptive non-traditional search procedure, has also been employed here to find the global optimum for the multi-variable objective function given in Equation(3.28). The initial *temperature* (T_o) has been taken to a very high value such as $500K$. To make the search procedure more robust in every coordinate directions of the hyper-dimensional search space, the number of iterations before temperature reduction (N_T) has been taken as 100 and number of cycles (N_s) performed in every iteration is taken as 20. As a result, before any temperature reduction the number of function evaluation required is

36000. Temperature is reduced further after thermal equilibrium has been reached at any particular temperature. The search is carried out until all the parameters finely converged to global optimum.

4.5 COMPUTATION ENVIRONMENT

In order to satisfy the requirements of the highly computation intensive nature of this problem a number of machines were used in this study. Some tasks were done by using the linux version of *FORTRAN 77* (f2c)compiler under *Linux environment* in pentium machines and rest of the job is done using *f77* -MIPS and MIPSpro F77 compiler in Silicon Graphics workstation with central processing units : MIPS R10000, Rev 2.6 under IRIX64 Release6.4 operating system. In both the cases, it took the cpu time 18000.00 sec on an average for a single run until converged, where all the eighteen decision variables have been considered to form the objective function.

Chapter 5

Results and Discussion

5.1 Results Obtained by Steepest Descent Method

The results obtained by using *steepest descent method* are given in Table A.6. It is rather well known that the gradient based traditional methods are very much sensitive to proper initial guess vector and also dependent upon the upgradation technique of the penalty parameters. Hence, *steepest descent method* is unable to converge properly to the global optimum for a non-linear, multi-modal objective function like Equation (3.28) used in this study. In this case it was not possible to optimize all the operational parameters simultaneously using the steepest descent search procedure.

5.2 Results Obtained by Genetic Algorithm

In this study *genetic algorithms* was successfully applied to all the three formulations described in Chapter 3. While operating with *genetic algorithms*, the task of optimization has become much easier compared to *steepest descent* method.

Using GA, we first tried to find out the optimum values for design parameters related to the oscillatory mold. The results obtained using *SGA* for the first formulation (Equation (3.5)) are given in Table A.5. In order to find out the relational behavior of the decision parameters with casting speed, we have varied the variable ranges in a predefined way.

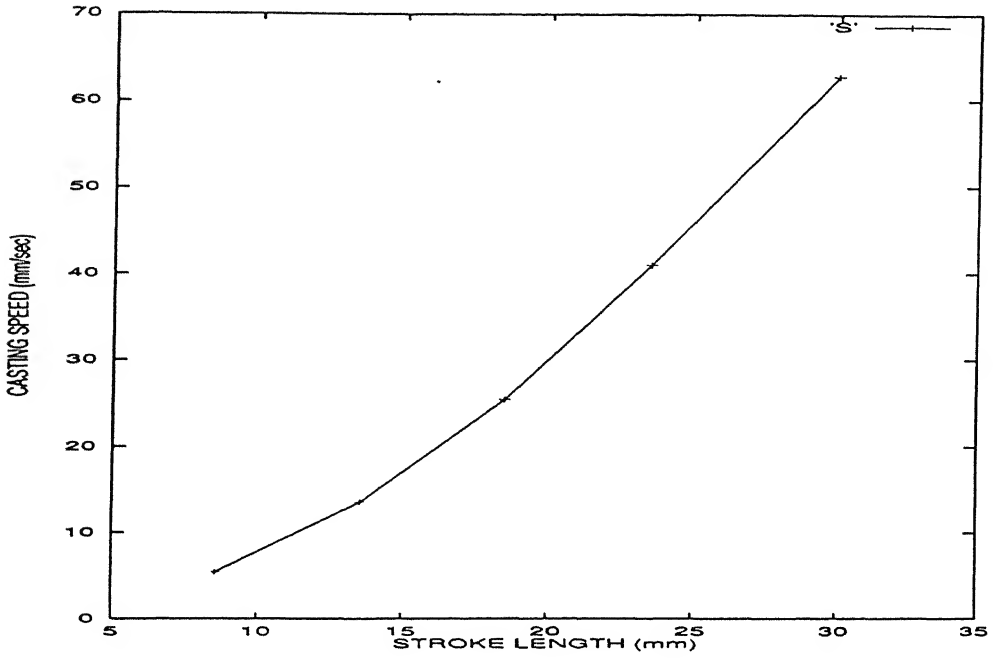


Figure 5.1: Optimum casting speed as a function of the variation of stroke length of the mold.

From Figures 5.1 and 5.2 we can easily predict that the casting speed increases with increase in stroke length and decrease in vitrification ratio of mold flux in oscillatory mold. The optimum values for different parameters obtained using the objective function given in Equation 3.28 are given in Tables A.8 through A.10.

5.3 RESULTS OBTAINED BY DIFFERENTIAL EVOLUTION AND SIMULATED ANNEALING

It took *differential evolution* atleast 1500 generations to reach the near optimal solution when all the eighteen parameters in the multi-variable objective function formulation described in Equation (3.28) were considered. Often it was necessary to run upto 2500 generations to obtain the proper convergence. Different optimum parameter values computed using *differential evolution* are enlisted in Tables A.11 through A.13.

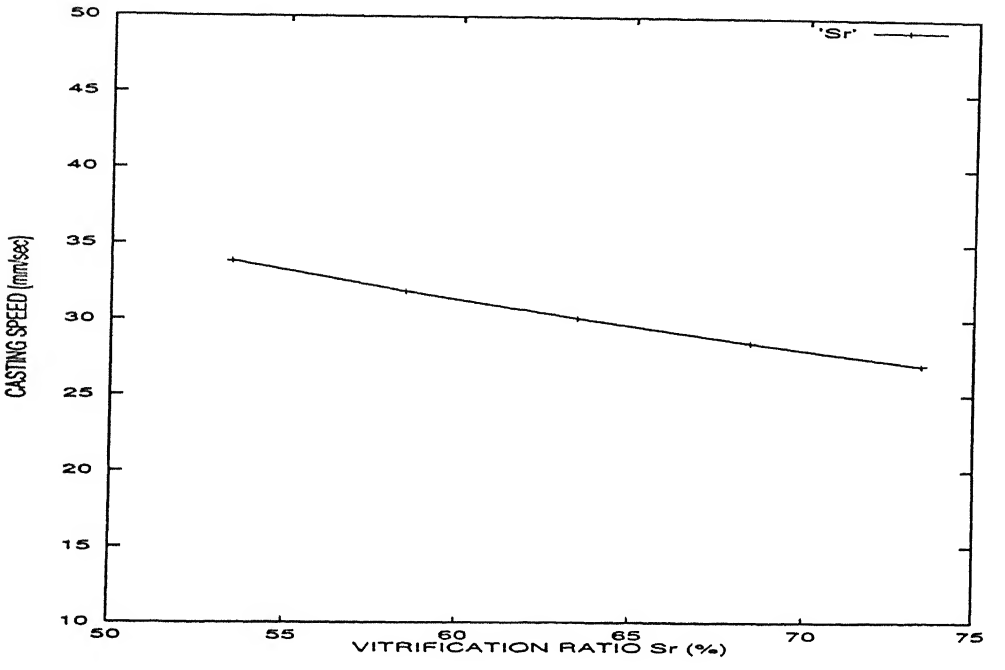


Figure 5.2: Optimum casting speed as a function of the variation vitrification ratio of the mold.

All the eighteen optimal decision parameter values obtained by *simulated annealing* itself, are given in Tables A.14 through A.16.

5.4 COMPARISON OF PERFORMANCES OF DIFFERENT OPTIMIZATION METHODS

The total number of function evaluations required in *simulated annealing* is 72,00,000 on an average for every single run until convergence. For *genetic algorithms* the corresponding number is much less (approximately 6,75,000) which is also comparable with function evaluations required in *differential evolution.*

In order to compare the performance of all these optimization procedures adopted here, the following figures (Figures 5.3-5.8) have been plotted, and they are also compared with the industrial data [51] simultaneously. The optimum casting speed has been calculated by

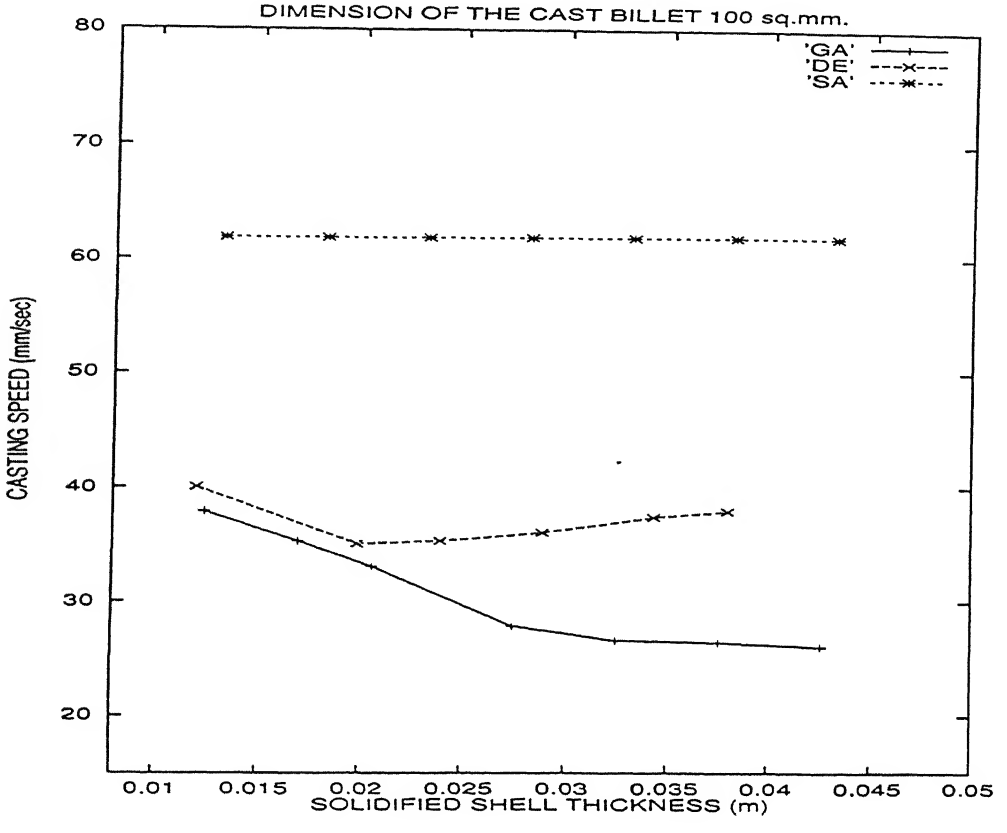


Figure 5.3: Optimum casting speed as a function of the variation of solidified shell thickness in different optimization procedures.

these methods for different dimension of the cast billets.

From the figures, it is clear that the predictions made by *genetic algorithms* for the optimum value of casting speed as a function of different operational parameters are more satisfactory compared to *differential evolution*, and *simulated annealing*. The results obtained by *simulated annealing* for the optimal casting speed with the variation of solidified shell thickness are physically infeasible showing no variation in casting speed. The results obtained by traditional steepest descent method are not fully converged, and the values of the equality constraint as well as the optimum casting speed are so large that they are not considered while drawing the plots. Also the optimum casting speed predicted by this method is so high that it appears practically infeasible. The variation of casting speed with negative strip time is also considered. *Simulated annealing* gives much higher values than the normal

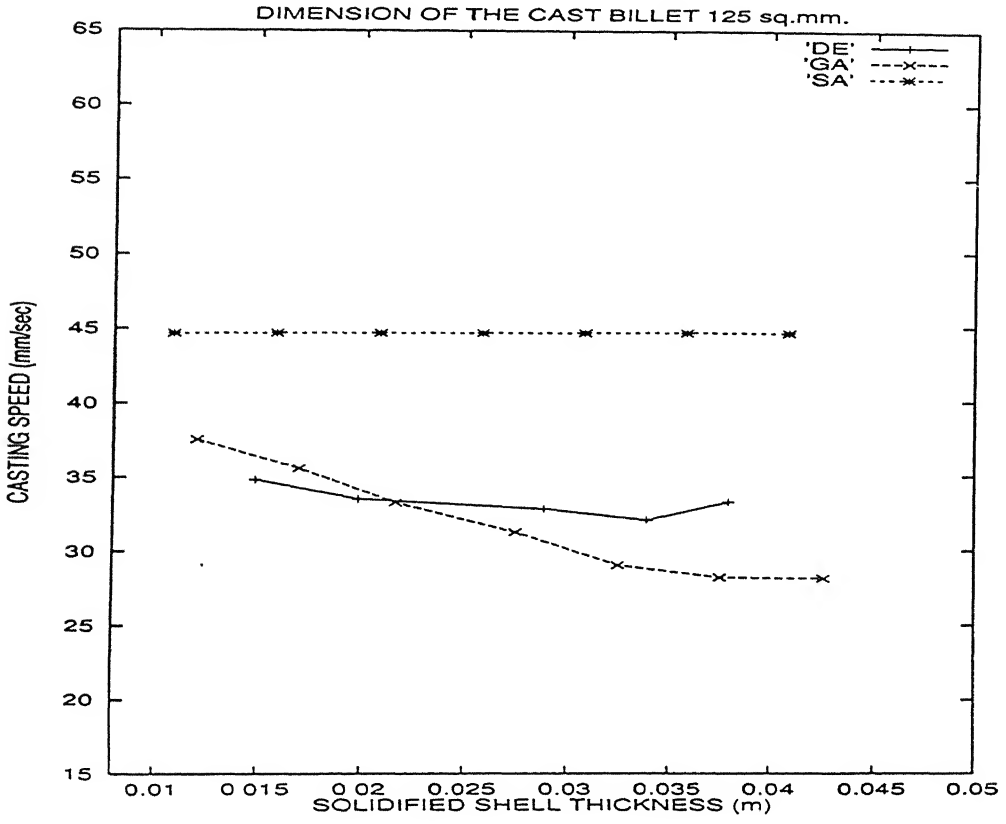


Figure 5.4: Optimum casting speed as a function of the variation of solidified shell thickness in different optimization procedures.

industrial practice, whereas *genetic algorithms* and *differential evolution* predicts the mean approximately. It can be concluded therefore that *genetic algorithms* itself work better than all the other optimization procedures.

5.5 THE COMBINED OPTIMIZATION APPROACH BASED ON GA AND SA

In the plots, the results obtained by *simulated annealing* are quite interesting, because the value of the equality constraint (Equation (3.34)) obtained there almost tends to zero $\{O(10^{-6})\}$, and simultaneously all the inequality constraints are satisfied, whereas *genetic*

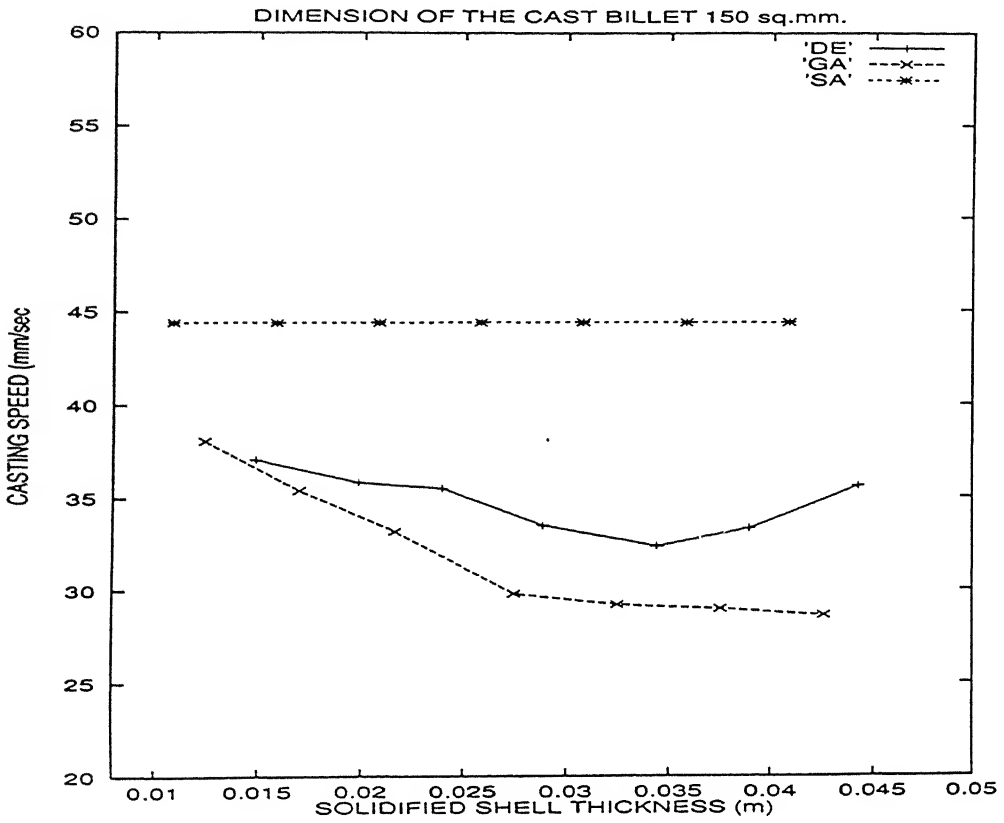


Figure 5.5: Optimum casting speed as a function of the variation of solidified shell thickness in different optimization procedures.

algorithms itself is unable to satisfy constraints to that level. Therefore, in order to obtain the global optimum values of the casting speed with the variation of other design variables we have used *genetic algorithms* to reach the near optimal point and for finer convergence *simulated annealing* is used with a low initial temperature (T_o). The results obtained by the optimization scheme adopted here are given in Tables A.17 through A.19.

The results which are obtained by coupling SGA and SA are almost comparable with the industrial practice. The velocity profile that has been obtained with the variation of solidified shell thickness properly matches with the normal operational trends [51]. The cooling water flow rate that has been taken in the calculation is approximated form the value given by Gieger and Poirer [49].

The dependence of casting speed on liquid pool depth (Y_p) and flux pool depth (D) is

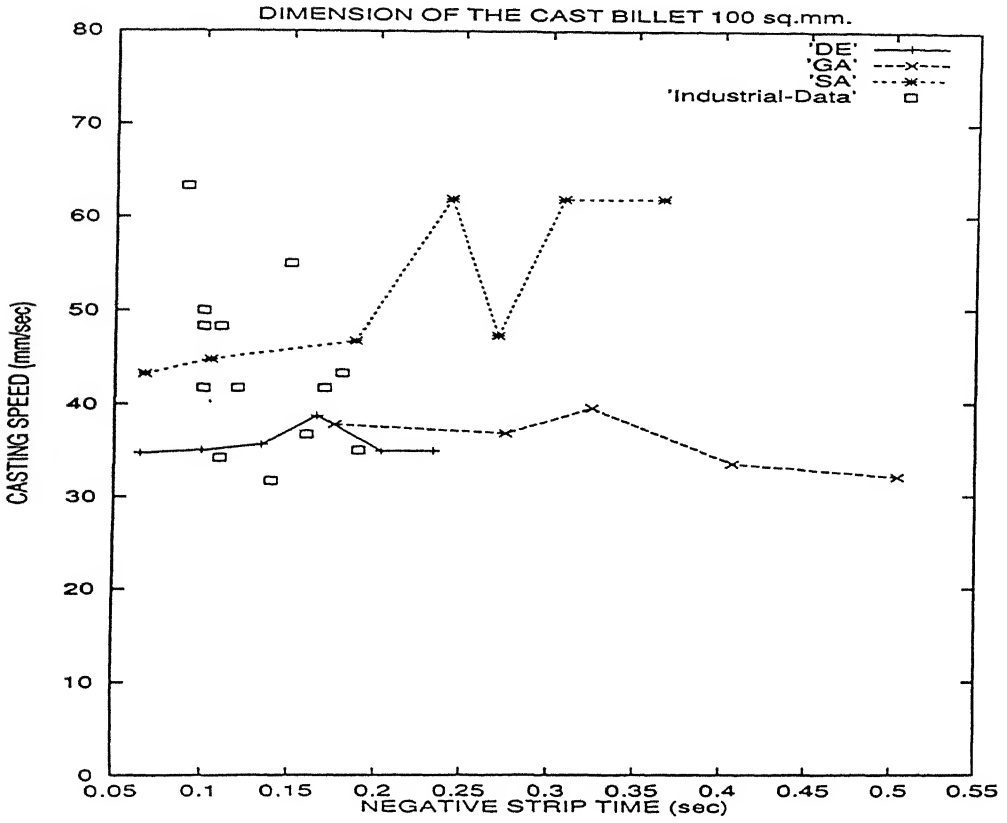


Figure 5.6: Optimum casting speed as a function of the variation of negative strip time in different optimization procedures.

also shown in the Figure 5.9, from which it can be concluded that the casting speed does not vary so much with these parameters in the range. However, there is a significant effect of stroke length (S) of the oscillatory mold on the casting speed which is shown in Figure 5.11. The casting speed increases proportionately with the increase in stroke length. The casting speed varies inversely with the vitrification ratio (S_R). The casting speed also increases with an increase in frequency of oscillation, but after a certain range the casting speed almost become a constant. In the range of parameters considered in this study, the contribution of the term $\frac{10.0 S_R N}{abdwf}$ in the denominator of Equation (3.5) appears to be significant. The values of the optimized velocity therefore goes up with increasing billet dimension. All these trends are shown in Figures 5.9-5.12.

The negative strip time (N) shows a typical relationship with casting speed. Initially

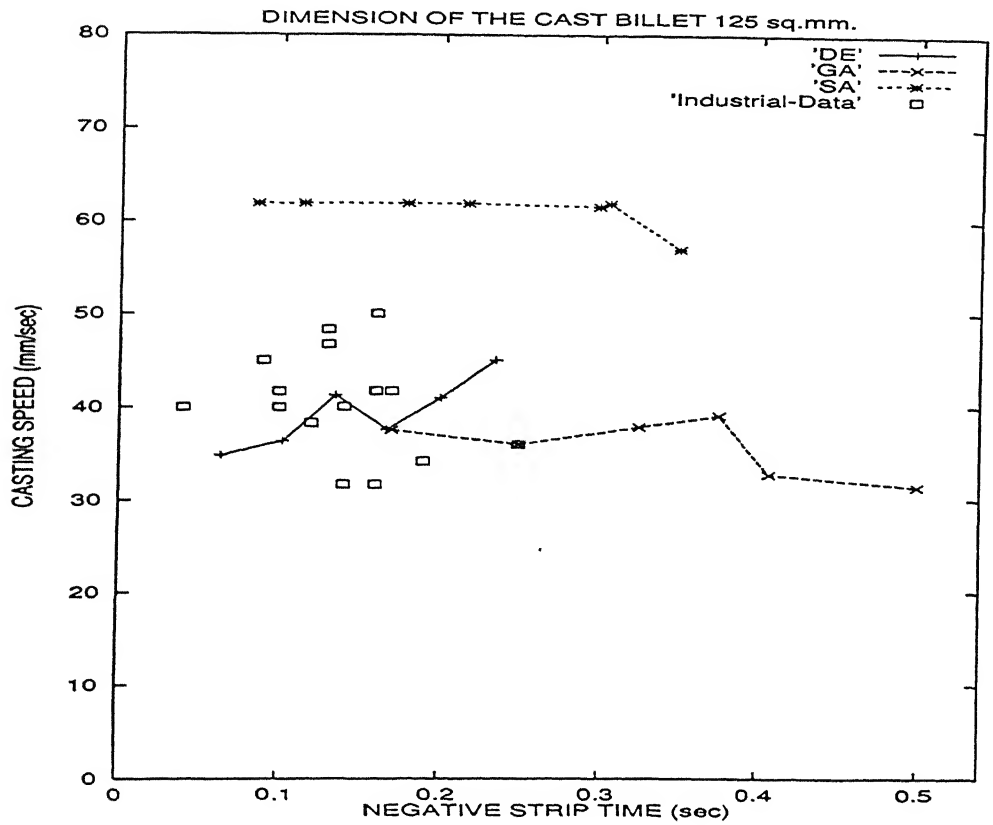


Figure 5.7: Optimum casting speed as a function of the variation of negative strip time in different optimization procedures.

the casting speed increases with the negative strip time, but with further increase in N the casting speed decreases. It can be concluded that the solidified shell thickness, vitrification ratio, negative strip time, stroke length are the major important parameters in controlling the casting speed, which almost remains constant for a reasonable variation of surface temperature T_s between 1450 to 1500 K and inner mold wall temperature T_o between 400 to 450 K, because of the other parameter settings.

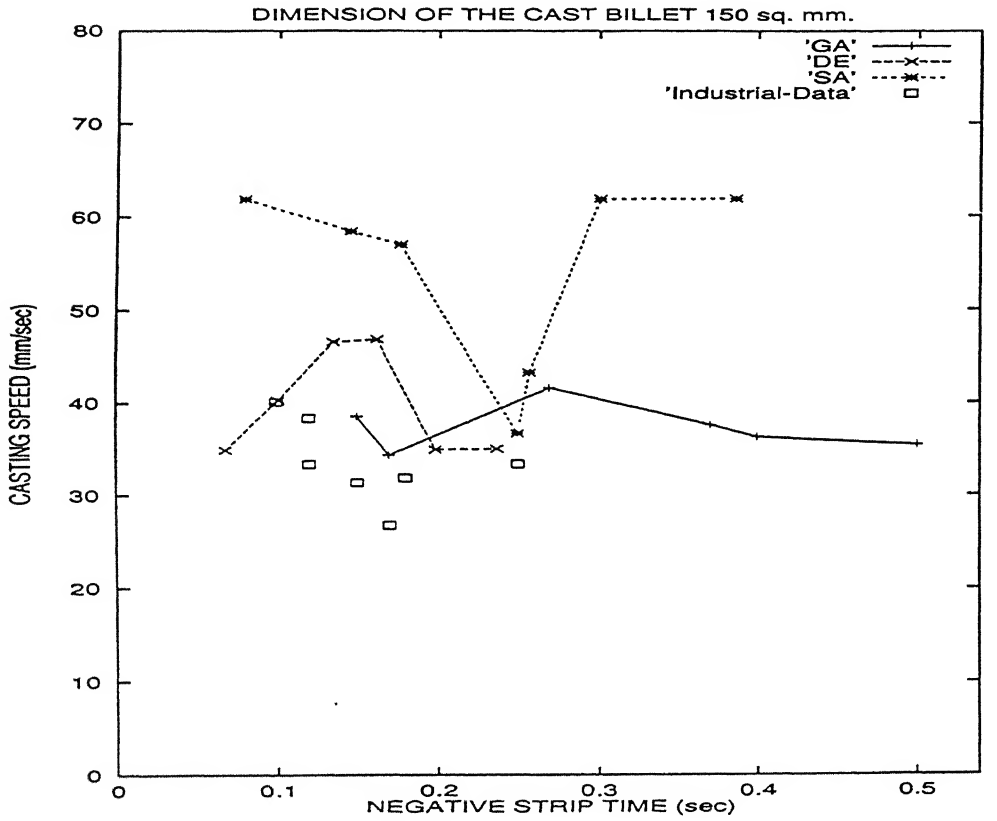


Figure 5.8: Optimum casting speed as a function of the variation of negative strip time in different optimization procedures.

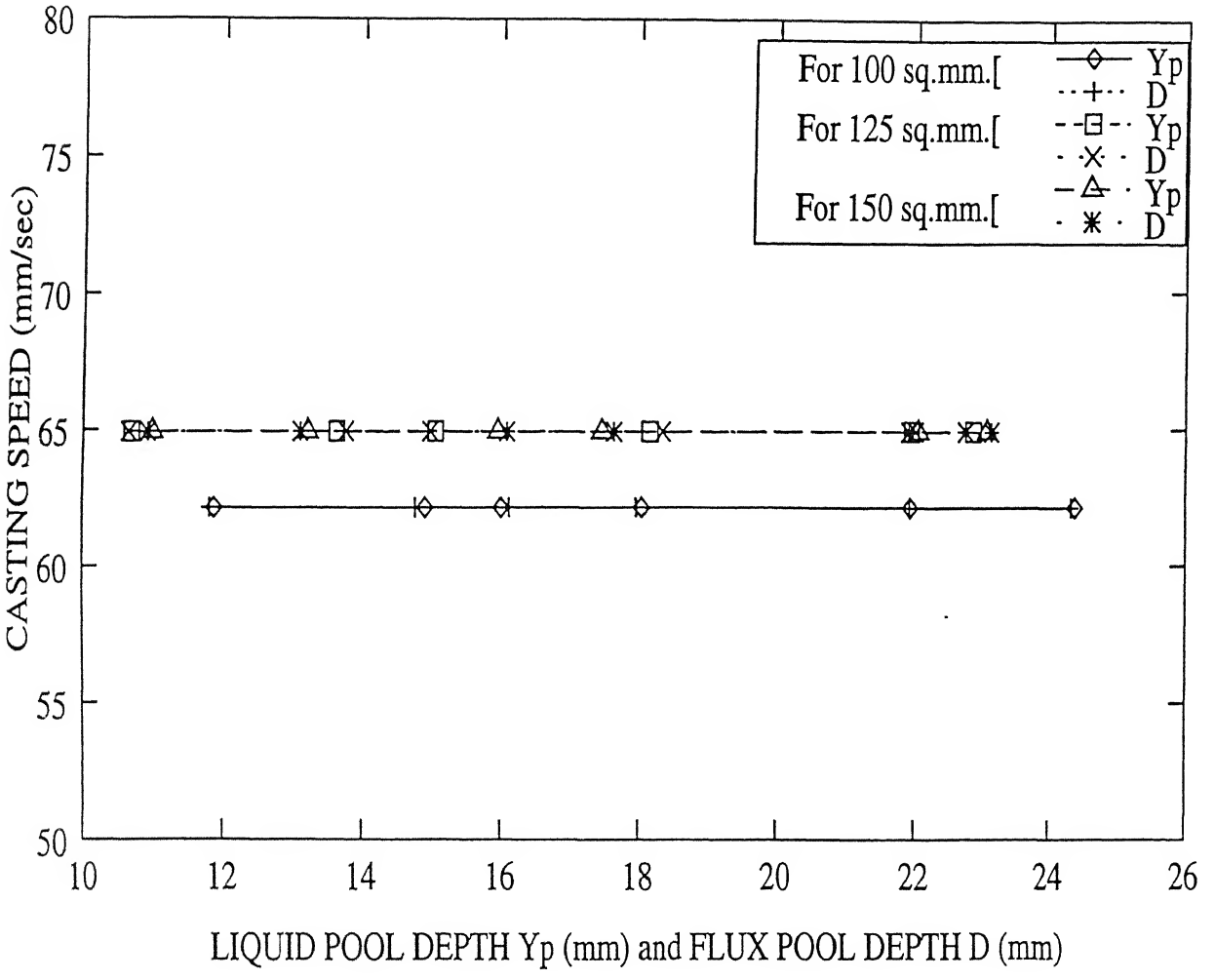


Figure 5.9: Optimum casting speed as a function of the variation of flux pool depth and liquid pool depth, when the dimension of the cast billet considered as 100 sq.mm., 125 sq.mm., and 150 sq.mm.

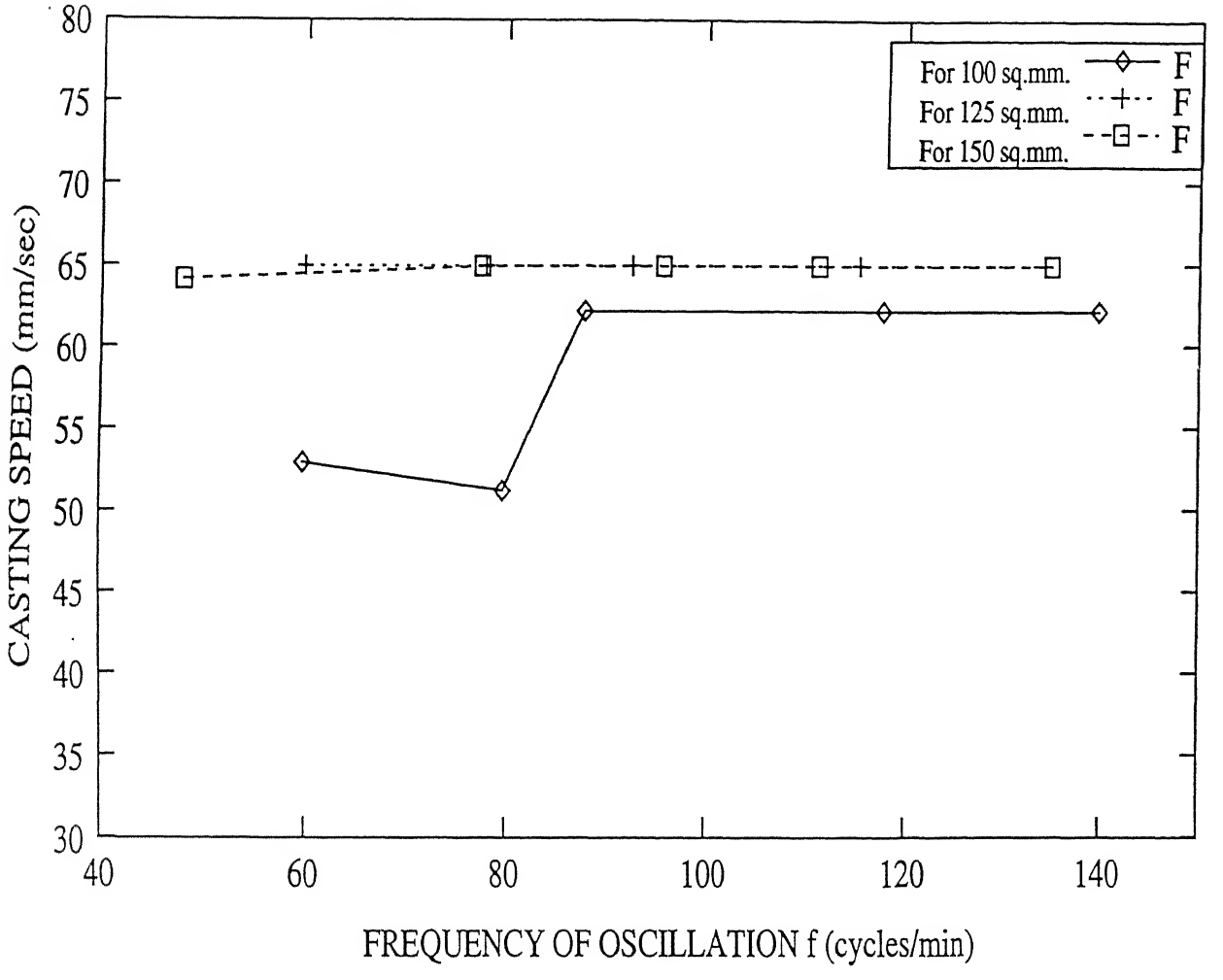


Figure 5.10: Optimum casting speed as a function of the variation oscillatory mold frequency, when the dimension of the cast billet considered as 100 sq.mm., 125 sq.mm., and 150 sq.mm.

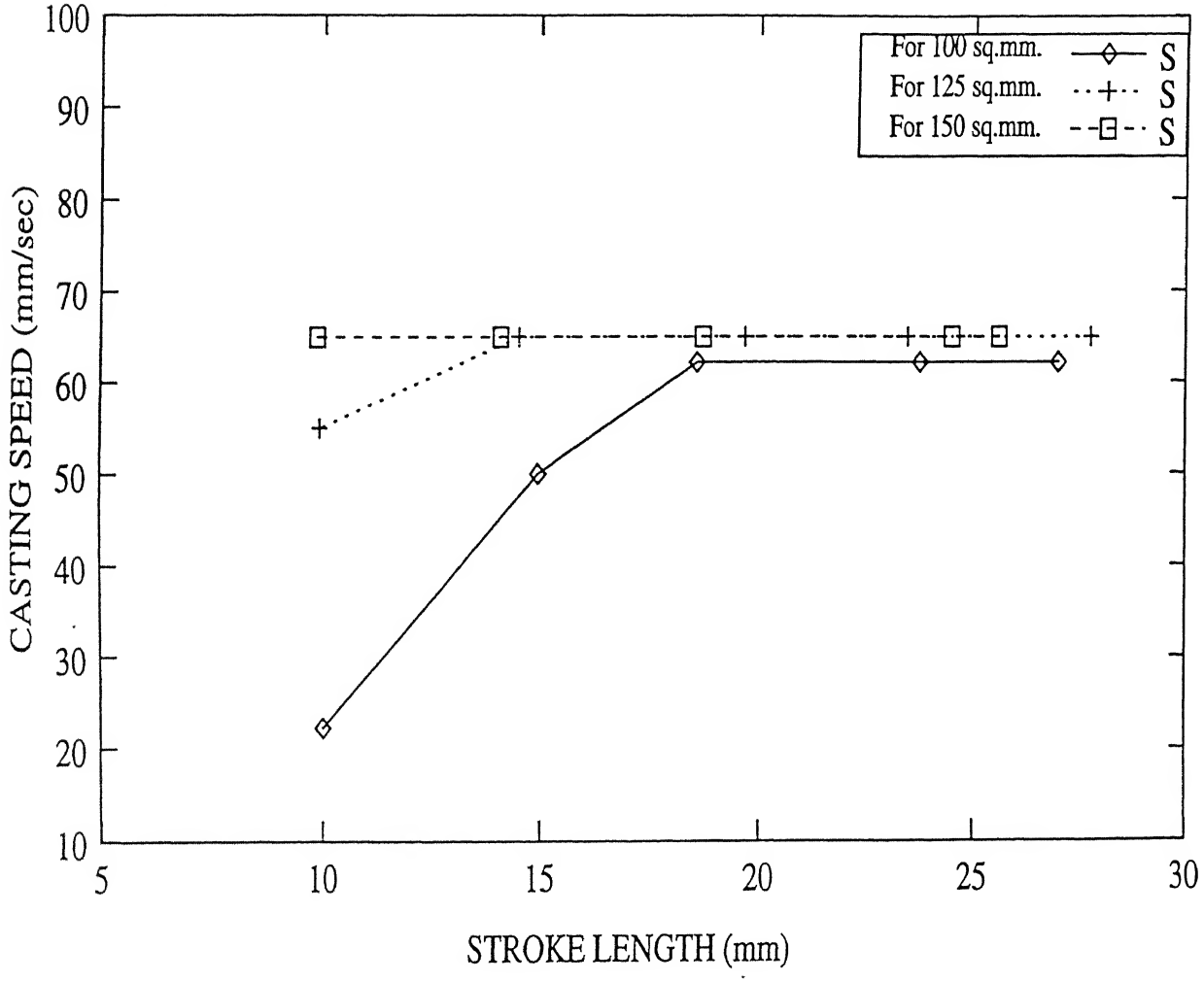


Figure 5.11: Optimum casting speed as a function of the variation oscillatory mold frequency, when the dimension of the cast billet considered as 100 sq.mm., 125 sq.mm., and 150 sq.mm.

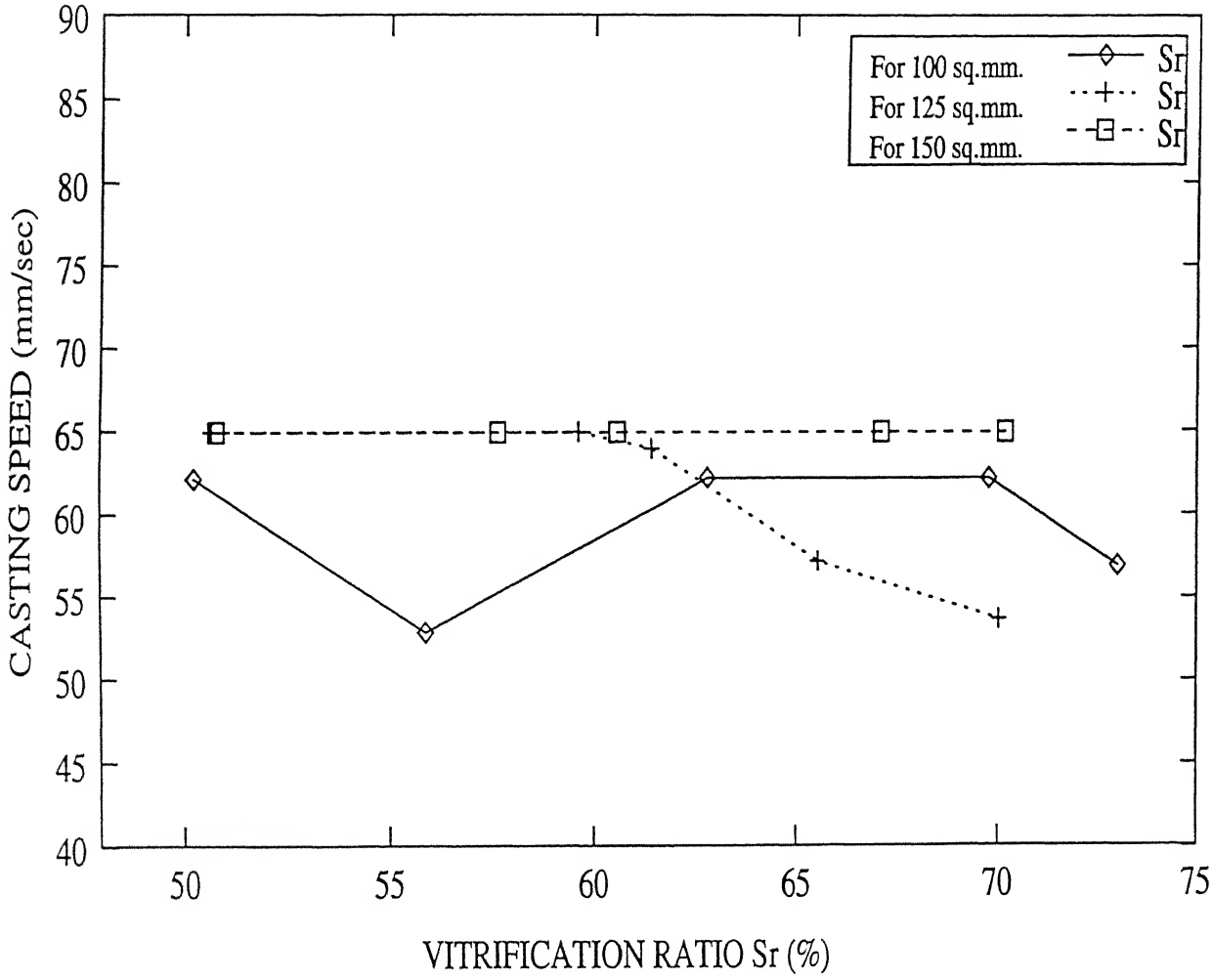


Figure 5.12: Optimum casting speed as a function of the variation oscillatory mold frequency, when the dimension of the cast billet considered as 100 sq.mm., 125 sq.mm., and 150 sq.mm.

Chapter 6

Conclusions

A large number of continuous casting process variables have been rigorously optimized in this study to yield the optimum casting speed which has not been attempted before for this process. The relational behavior of optimum casting speed with different operational parameters of billet casting obtained in this study matches well with the industrial practice. The optimum casting speed generally increases with the increase in stroke length and decreases with the increase in vitrification ratio. The effect of liquid pool depth and flux pool depth on the optimum casting speed does not appear to be very significant. For different dimensions of billets considered in this study, the basic trends of casting speed variation remain the same. The results of this study indicates that the casting speed increases with an initial increase in negative strip time, but later it decreases when the strip time is sufficiently high. The casting speed decreases rapidly with the increase in solidified shell thickness, but eventually the rate of decrease slows down. The optimum casting speed varies proportionately with the frequency of mold oscillation for every range of billet dimensions considered in this study. It can be concluded that the solidified shell thickness, negative strip time, stroke length of mold are the major factors influencing the casting speed.

Among different optimization procedures adopted in this study, *genetic algorithms* are found to be the most robust. From the results it appears that *simulated annealing* has predicted higher casting velocity than the other optimization methods for certain operational conditions. However in those cases, though the value of the equality constraint almost tends to zero, some correction factors applied in this model are not properly satisfied within their variable bounds, rendering those high casting velocities infeasible. On the other hand, for

all the operating conditions *genetic algorithms* are able to predict the near optimal values comparable with the industrial practice in most of the cases. Furthermore, its efficiencies in terms of the number of function evaluations and search power are found to be much better than all the other optimization techniques used here.

In order to handle more than one objective functions, *the method of objective weighing* is not very efficient compared to *Non-dominated Sorting Genetic Algorithms* with a *Pareto optimal formulation*. Also, the code needs to be parallelized in order to lower the computation time, which can be undertaken during the subsequent phases of this research work.

For a further detailed analysis of the continuous casting mold, the model used in this study needs to be refined further, considering the non-linear temperature profiles within and outside the cast billet, the gap between the strand and the mold wall etc. In order to understand the heat transfer mechanism in details, it is necessary to consider that the gap between the mold wall and the strand which is infiltrated with the crystalline solid mold powder as well as molten flux. The viscosity of the molten flux pool has also a significant effect on the casting speed, which varies with temperature distribution in the mold.

In addition to the eighteen parameters considering in this study, few more may be necessary to predict the operational variables accurately. Some of the important parameters which are not being included here are shape of mold taper, the corner radius of the mold, composition of the steel being cast, different mold flux composition under different operational conditions, and the variation of kinematic viscosity of the mold flux with temperature-all of which need to be considered in a further refined model. The steady-state heat extraction pattern in the mold which is considered to be linear in this study may not be the proper approximation of the true heat extraction pattern. Because the gap between the mold wall and solidified strand is infiltrated with a layer solid flux film and liquid flux film, whose thicknesses are varying continuously through out the operation resulting in a non-linear temperature profile. The convective heat transfer due to the downward motion of the strand is not very preciously considered. Due to the corner radius of the mold the heat transfer may not be uniform and even all through out the mold, which also needs to be considered in the formulation of the objective functions. Finally the mold shape and number of tapers can always be varied from one operational condition to another, adding further challenges to a rigorous optimization of the process.

Appendix A

TABLES

Table A.1: The value of the constants

PARAMETER	VALUE	REF. NO.
$k^{'}$	49e-03 $(KJm^{-1}s^{-1}K^{-1})$	[3, 51]
k	374e-03 $(KJm^{-1}s^{-1}K^{-1})$	[3, 51]
k_m	2e-03 $(KJm^{-1}s^{-1}K^{-1})$	[3, 51]
k_w	6.34e-03 $(KJm^{-1}s^{-1}K^{-1})$	[3, 50]
h	1.42 $(KJm^{-1}s^{-1}K^{-1})$	[3, 49]
$C_p^{'}$	0.670 $(KJKg^{-1}K^{-1})$	[3, 49]
C_{PL}	0.754 $(KJKg^{-1}K^{-1})$	[3, 49]
C_P	0.4187 $(KJKg^{-1}K^{-1})$	[3, 51]
C_{PW}	4.174 $(KJKg^{-1}K^{-1})$	[3, 50]
H_f	0.2675 $(KJKg^{-1})$	[3, 49]
μ_w	6.524e-04 $(Kgm^{-1}s^{-1})$	[3, 50]
Δg	0.2e-03 (m)	[3, 51]
Δt_{mold}	35e-03 (m)	[3, 51]
w	100e-03 (m)	[3, 51]
$\rho^{'}$	7860 (Kgm^{-3})	[3, 51]
ρ	8940 (Kgm^{-3})	[3, 51]
ρ_w	992.2 (Kgm^{-3})	[3, 50]
T_M	1772 (K)	[3, 49]
T_p	1822 (K)	[3, 49]

Table A.2: The variable bounds for the first objective function

VARIABLES	LOWER BOUND	UPPER BOUND
Y_p (mm)	10.0	20.0
D (mm)	10.0	20.0
S (mm)	5.0	30.0
F (cpm)	15.0	140.0
δ (mm)	4.0	12.0
N (sec)	0.05	0.6
Sr (%)	50.0	75.0
w $(Kgton^{-1})$	0.2	0.9

Table A.3: The variable bounds for the second objective function

VARIABLES	LOWER BOUND	UPPER BOUND
$T_M - T_O$ (K)	800.0	1600
L (m)	0.8	1.2
M (m)	0.008	0.011
M (m)	0.011	0.014
M (m)	0.014	0.017
M (m)	0.017	0.020
M (m)	0.020	0.023
M (m)	0.023	0.026
M (m)	0.026	0.030
M (m)	0.030	0.033
M (m)	0.033	0.036
M (m)	0.036	0.040
M (m)	0.040	0.045

Table A.4: The variable bounds for third objective function

VARIABLES	LOWER BOUND	UPPER BOUND
T_O (K)	400.0	700.0
T_s (K)	1200	1500
T_{Wout} (K)	302	320
ϵ	0.0	0.5
$\delta_1(KJm^{-2}s^{-1}K^{-1})$	-2.5	0.5
$\delta_2(KJm^{-2}s^{-1}K^{-1})$	-1.8e-03	1e-03
$\beta(KJm^{-2}s^{-1})$	1e01	1.85e05
$\phi(ms^{-1})$	5	15
M (m)	.008	.011
M (m)	.011	.014
M (m)	.014	.017
M (m)	.017	.020
M (m)	.020	.023
M (m)	.023	.026
M (m)	.026	.030
M (m)	.030	.035
M (m)	.035	.040
M (m)	.040	.045

Table A.5: Results obtained for first objective function by GA(the dimension of billet 150sq.mm.)

RUN NO. #	Y_P mm	D mm	S mm	F cpm	δ mm	N sec	Sr %	w (Kg/ton)	VELOCITY m/min
1	14.545	14.70	22.458	119.19	10.278	.370	56.79	.858	8.273
2	15.644	15.81	24.548	114.73	7.968	.382	50.81	.798	7.434
3	19.391	19.69	24.661	118.87	10.965	.304	67.90	.756	6.815
4	17.27	17.09	8.5505	58.249	11.856	.379	67.20	.826	.323
5	17.27	17.09	13.551	58.249	11.856	.379	67.20	.826	.8132
6	17.27	17.09	18.551	58.249	11.856	.379	67.20	.826	1.524
7	17.27	17.09	23.551	58.249	11.856	.379	67.20	.826	2.456
8	16.784	16.60	29.968	56.267	9.3969	.239	55.20	.898	3.770
9	17.27	17.09	19.78	58.249	11.856	.379	53.44	.826	2.031
10	17.27	17.09	19.78	58.249	11.856	.379	58.44	.826	1.911
11	17.27	17.09	19.78	58.249	11.856	.379	63.44	.826	1.805
12	17.27	17.09	19.78	58.249	11.856	.379	68.42	.826	1.710
13	17.27	17.09	19.78	58.249	11.856	.379	73.42	.826	1.62
14	16.092	15.92	23.884	23.267	11.801	.328	54.58	.778	.560
15	16.092	15.92	23.884	38.267	11.801	.328	54.58	.778	1.394
16	16.092	15.92	23.884	53.267	11.801	.328	54.58	.778	2.503
17	16.092	15.92	23.884	68.267	11.801	.328	54.58	.778	3.828
18	16.092	15.92	23.884	83.267	11.801	.328	54.58	.778	5.329
19	16.092	15.92	23.884	98.267	11.801	.328	54.58	.778	6.974
20	16.092	15.92	23.884	113.26	11.801	.328	54.58	.778	8.739
21	16.092	15.92	23.884	46.535	11.801	.176	54.58	.778	1.873
22	16.092	15.88	23.991	54.035	11.301	.276	61.76	.894	2.525
23	16.092	15.88	23.919	54.035	11.031	.376	61.76	.895	2.462
24	17.186	16.97	23.644	58.707	4.6216	.404	54.57	.859	2.0504
25	17.186	16.98	23.644	59.000	4.6216	.504	54.57	.862	1.9203

Table A.6: Results obtained by the Steepest Descent Method

RUN #	T_p K	L m	β	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	ϕ	M m	Cast Speed (m/sec)
1	1822.0	1.1	.118e06	414.7	1499.9	309.5	0.287	0.287	-.114	14.79	0.008	0.5015
2	1822.0	1.1	.982e05	415.9	1208.6	302.3	0.131	0.131	-1.10	5.48	0.011	0.3471
3	1822.0	1.1	.979e05	460.6	1472.5	316.0	0.325	0.325	-1.90	5.76	0.014	0.2663
4	1822.0	1.1	.982e05	409.0	1399.4	313.5	0.430	0.430	-2.86	5.86	0.017	0.2161
5	1822.0	1.1	.980e05	414.9	1459.5	315.0	0.189	0.058	-4.11	10.10	0.020	0.1547
6	1822.0	1.1	.101e06	412.1	1459.2	316.2	0.343	0.356	-5.75	9.06	0.023	0.1363
7	1822.0	0.8	.980e05	421.4	1209.0	319.6	0.486	0.490	-1.14	5.65	0.011	0.2865
8	1822.0	1.5	.118e06	530.5	1207.6	309.0	0.467	0.460	-1.22	5.20	0.011	0.5092
9	1822.0	1.9	.118e06	690.5	1440.8	310.3	0.475	0.470	-1.29	5.12	0.011	0.5997
10	1822.0	2.5	.117e06	568.5	1214.5	315.7	0.484	0.483	-1.63	8.68	0.011	0.8406

Table A.7: Results obtained for the second objective function by GA

RUN #	Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/ton	$(T_M - T_O)$ K	L m	M m	VEL. m/min
1	11.772	11.643	24.765	73.995	11.913	.1	51.234	.849	855.82	.93831	.009	7.41
2	14.40	14.563	24.272	72.369	11.643	.232	56.814	.896	801.42	.98942	.0138	4.97
3	14.256	14.106	24.856	71.508	9.7422	.266	52.236	.84392	992.65	.91836	.016	4.37
4	15.076	15.269	24.818	70.644	10.577	.317	57.729	.87173	1106.0	.996	.018	4.36
5	17.194	17.380	24.800	71.525	10.492	.417	63.398	.76045	1003.5	1.041	.021	3.48
6	18.645	18.848	24.737	73.053	6.7133	.419	66.980	.7437	842.54	1.0388	.023	2.81
7	19.572	19.806	24.711	72.628	6.7457	.446	55.868	.55414	963.49	.98871	.026	2.56
8	16.707	16.894	23.614	67.256	8.7116	.284	65.548	.82452	973.8	.94695	.030	2.33
9	17.235	17.425	23.952	74.293	9.2129	.494	64.895	.51563	906.9	.93860	.033	2.27
10	16.483	16.688	23.592	64.082	5.7286	.255	50.279	.6472	1099.3	.99520	.037	2.12

Table A.8: Results obtained for the multi-objective function by GA (for dimension of cast billet 100 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{lVout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
15.00	14.90	24.50	60.8	10.0	0.12	55.80	0.854	475	1460	303	0.25	0.20	-.0030	.11e06	9.89	0.012	1.05	32.00
15.00	14.90	23.98	58.0	10.5	0.12	58.00	0.890	415	1450	303	0.35	0.34	-.0016	.98e05	10.20	0.017	1.10	30.52
14.50	14.60	22.98	55.8	10.5	0.11	60.00	0.750	475	1475	305	0.30	0.27	-.0024	.98e05	11.50	0.022	1.10	27.91
14.20	14.30	22.98	55.1	10.9	0.12	65.00	0.851	480	1470	306	0.29	0.28	-.0056	.11e06	9.75	0.027	1.04	27.54
14.00	14.06	21.01	55.0	10.5	0.12	60.50	0.875	489	1465	305	0.28	0.29	-.0016	.98e05	10.01	0.032	1.04	26.66
14.50	14.60	20.90	55.7	11.0	0.10	58.90	0.789	480	1480	303	0.27	0.28	-.0020	.97e05	10.20	0.037	1.04	26.48
14.80	15.01	19.98	55.0	10.7	0.18	51.00	0.850	460	1460	305	0.25	0.24	-.0016	.104e06	10.50	0.042	1.03	27.78
15.01	15.98	24.90	52.1	11.0	0.17	55.00	0.908	425	1460	303	0.32	0.27	-.0016	.98e05	9.80	0.018	1.09	31.27
16.50	16.40	25.00	56.8	10.3	0.27	60.80	0.981	415	1470	306	0.27	0.05	-.0018	.97e05	10.80	0.018	1.04	32.32
15.90	16.00	25.01	55.9	10.5	0.32	65.00	0.879	510	1460	305	0.25	0.24	-.0018	.11e06	9.98	0.020	1.10	31.86
17.09	16.90	24.90	60.2	10.9	0.37	58.00	0.809	511	1460	304	0.34	0.36	-.0017	.101e06	10.20	0.020	1.10	32.04
17.50	17.40	24.80	58.0	9.80	0.41	58.50	0.890	480	1450	303	0.43	0.41	-.0016	.98e05	10.50	0.020	1.10	29.58
17.50	17.40	24.70	56.0	10.1	0.50	60.00	0.900	480	1440	303	0.41	0.38	-.0023	.98e05	9.50	0.020	1.05	26.73

Table A.9: Results obtained for the multi-objective function by GA (for dimension of cast billet 125 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
15.08	14.89	23.89	59.8	10.1	0.12	54.57	0.859	475	1450	306	0.25	0.21	-.0032	.106e06	10.11	0.012	1.05	34.53
14.08	14.1	23.75	58.9	11.1	0.12	59.01	0.798	415	1445	305	0.32	0.32	-.0014	.98e05	9.61	0.017	1.10	35.58
14.50	14.61	22.89	57.8	10.8	0.12	61.01	0.901	498	1470	305	0.27	0.27	-.0034	.98e05	10.31	0.022	1.10	34.27
14.40	14.20	21.67	54.8	10.9	0.12	62.50	0.890	489	1470	305	0.34	0.29	-.0045	.11e06	9.87	0.027	1.04	29.06
13.80	13.98	21.01	52.1	10.9	0.19	60.80	0.925	490	1460	304	0.24	0.21	-.0014	.98e05	10.08	0.032	1.05	31.27
14.50	14.60	19.98	53.9	10.3	0.10	56.87	0.891	470	1461	303	0.31	0.25	-.0025	.97e05	9.90	0.037	1.04	27.16
15.10	15.00	19.10	54.0	10.0	0.11	62.10	0.870	460	1471	305	0.21	0.21	-.0050	.102e06	10.90	0.042	1.03	27.23
16.10	15.90	24.50	49.1	11.0	0.17	55.02	0.802	415	1450	303	0.32	0.29	-.0014	.98e05	8.61	0.017	1.09	32.30
16.10	15.92	24.98	54.0	10.3	0.27	60.98	0.894	416	1461	306	0.28	0.05	-.0015	.97e05	10.80	0.018	1.04	34.57
16.09	15.89	24.01	55.9	10.4	0.32	61.76	0.945	514	1495	306	0.24	0.01	-.0016	.106e06	9.81	0.020	1.10	35.68
17.08	17.01	24.24	58.9	11.0	0.37	56.89	0.980	512	1460	306	0.34	0.35	-.0015	.101e06	9.85	0.020	1.10	39.11
17.98	17.87	24.01	56.1	10.1	0.41	58.93	0.798	487	1410	303	0.42	0.40	-.0015	.98e05	9.40	0.020	1.10	31.34
17.60	17.67	24.10	56.0	10.9	0.50	59.80	0.897	478	1410	303	0.42	0.40	-.0015	.98e05	9.40	0.020	1.05	30.75

Table A.10: Results obtained for the multi-objective function by GA (for dimension of cast billet 150 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
15.18	14.98	23.64	58.7	10.6	0.12	54.57	0.859	472	1450	305	0.26	0.20	-.0023	.116e06	10.11	0.012	1.05	38.06
14.89	14.69	22.57	55.7	11.1	0.12	54.57	0.798	409	1445	303	0.35	0.36	-.0015	.98e05	9.61	0.017	1.10	35.40
14.87	14.79	21.39	54.8	10.8	0.11	58.23	0.825	489	1459	305	0.29	0.27	-.0024	.98e05	10.31	0.022	1.10	33.19
14.54	14.37	20.37	52.4	9.78	0.12	59.98	0.912	479	1465	306	0.30	0.28	-.0054	.101e06	8.97	0.027	1.04	29.12
13.97	14.12	20.01	50.1	10.3	0.11	57.80	0.845	487	1455	305	0.25	0.23	-.0014	.98e05	10.08	0.032	1.04	29.71
14.45	14.55	19.46	52.8	11.3	0.10	56.87	0.798	476	1456	303	0.30	0.21	-.0021	.97e05	9.83	0.037	1.04	28.97
14.95	15.01	18.98	54.9	9.84	0.10	58.92	0.897	456	1464	305	0.22	0.20	-.0045	.104e06	10.02	0.042	1.03	28.62
16.09	15.92	23.88	46.5	11.8	0.17	54.58	0.778	409	1414	303	0.35	0.36	-.0015	.98e05	8.61	0.017	1.10	34.28
16.09	15.88	23.91	54.0	11.3	0.27	61.76	0.894	414	1459	305	0.19	0.05	-.0015	.97e05	10.11	0.020	1.07	41.57
16.09	15.88	23.19	54.0	11.3	0.37	61.76	0.894	514	1459	306	0.25	0.01	-.0015	.116e06	9.81	0.020	1.10	37.58
17.18	16.98	23.64	58.7	10.6	0.40	54.57	0.859	512	1459	306	0.34	0.35	-.0015	.101e06	9.76	0.020	1.10	40.89
17.18	16.97	23.64	58.7	10.6	0.50	54.57	0.859	477	1390	303	0.49	0.43	-.0015	.98e05	9.40	0.020	1.10	35.39

Table A.11: Results obtained for the multi-objective function by DE (for dimension of cast billet 100 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
14.84	14.59	18.49	78.7	7.99	0.10	60.31	0.540	472	1400	305	0.20	0.25	0.0003	.95e05	10.11	0.012	1.04	29.91
14.46	14.25	22.59	85.2	8.55	0.10	59.29	0.603	688	1363	302	0.01	0.02	0.0003	.14e06	11.04	0.019	0.90	30.11
14.54	14.35	22.38	82.9	8.04	0.10	59.75	0.556	679	1375	302	0.02	0.07	0.0003	.13e06	10.58	0.024	0.90	28.49
14.99	15.12	22.99	81.1	8.35	0.09	59.63	0.681	665	1353	302	0.03	0.14	0.0003	.13e06	9.87	0.029	0.90	30.46
14.87	15.03	22.14	82.4	8.54	0.10	60.00	0.552	642	1365	302	0.03	0.18	0.0003	.13e06	10.21	0.034	0.92	29.47
14.98	14.96	20.58	78.7	7.81	0.10	59.82	0.568	611	1350	303	0.07	0.18	0.0004	.12e06	8.94	8.94	0.94	27.35
14.89	14.38	22.98	86.2	8.27	0.06	59.29	0.680	696	1363	302	0.01	0.01	0.0003	.14e06	10.91	0.015	0.90	28.18
15.00	15.54	22.96	88.9	8.50	0.09	58.45	0.640	691	1381	302	0.01	0.02	0.0003	.14e06	11.01	0.014	0.90	31.94
15.99	15.81	23.79	92.5	8.62	0.13	59.48	0.652	695	1361	302	0.01	0.01	0.0003	.14e06	11.61	0.015	0.90	35.22
14.30	14.27	24.18	90.6	8.19	0.16	59.29	0.649	694	1368	302	0.01	0.01	0.0003	.14e06	11.50	0.015	0.90	36.78
14.87	14.70	22.93	84.7	8.32	0.20	59.57	0.672	693	1385	302	0.01	0.01	0.0003	.13e06	10.65	0.015	0.90	34.00
14.67	14.79	21.61	88.5	7.87	0.23	59.60	0.640	694	1356	302	0.01	0.01	0.0003	.14e06	11.35	0.015	0.90	32.94

Table A.12: Results obtained for the multi-objective function by DE (for dimension of cast billet 125 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
14.87	15.14	18.47	58.6	8.38	0.10	58.95	0.540	693	1372	302	0.01	0.01	0.0003	.13e06	10.89	0.015	0.90	34.81
14.87	15.19	22.80	86.7	8.25	0.10	59.98	0.589	687	1366	302	0.01	0.04	0.0003	.13e06	10.59	0.019	0.90	33.51
14.81	14.69	22.95	90.4	8.32	0.09	60.53	0.685	681	1368	302	0.01	0.05	0.0003	.13e06	10.73	0.024	0.90	35.55
14.64	14.68	22.41	80.9	8.29	0.10	60.27	0.662	668	1362	302	0.03	0.14	0.0003	.13e06	10.34	0.029	0.90	32.83
14.94	14.43	21.36	84.3	8.64	0.10	58.94	0.560	630	1350	302	0.04	0.20	0.0003	.13e06	9.97	0.034	0.92	32.07
14.32	14.25	22.35	81.4	8.22	0.10	59.87	0.638	628	1357	302	0.06	0.21	0.0002	.12e06	9.69	0.038	0.90	29.92
14.75	14.85	21.90	85.4	8.83	0.06	59.44	0.690	694	1369	302	0.01	0.02	0.0003	.13e06	10.47	0.015	0.90	29.73
14.89	15.07	22.84	89.6	8.51	0.10	59.68	0.681	693	1368	302	0.01	0.02	0.0004	.13e06	10.54	0.015	0.90	35.70
15.67	15.47	23.62	94.4	9.23	0.13	59.01	0.636	694	1368	302	0.01	0.01	0.0003	.14e06	11.12	0.015	0.90	41.23
14.17	14.53	22.99	83.5	8.23	0.16	59.01	0.615	691	1364	302	0.01	0.02	0.0003	.13e06	11.10	0.015	0.90	37.59
14.91	15.28	23.77	85.6	8.05	0.20	59.64	0.651	696	1370	302	0.01	0.01	0.0003	.14e06	11.33	0.015	0.90	40.96
14.24	14.52	23.24	94.8	8.66	0.23	59.33	0.600	693	1364	302	0.01	0.02	0.0003	.13e06	10.23	0.015	0.90	45.00

Table A.13: Results obtained for the multi-objective function by DE (for dimension of cast billet 150 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ϵ	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
15.01	15.05	22.78	86.1	8.53	0.11	60.03	0.625	692	1368	302	0.01	0.01	0.0003	.14e06	11.49	0.015	0.90	37.06
14.65	14.75	22.70	81.9	8.29	0.10	59.39	0.677	687	1359	302	0.01	0.04	0.0003	.14e06	10.87	0.019	0.90	35.52
14.56	14.85	22.79	82.1	8.19	0.10	59.17	0.633	677	1371	302	0.02	0.07	0.0002	.13e06	10.04	0.024	0.90	35.52
14.76	15.01	22.50	81.3	8.14	0.10	60.36	0.750	657	1359	302	0.03	0.15	0.0003	.14e06	10.96	0.029	0.90	35.94
14.94	14.58	21.66	82.8	7.78	0.10	59.65	0.539	662	1364	302	0.03	0.14	0.0003	.13e06	10.32	0.034	0.90	32.37
14.39	14.87	21.20	79.0	8.25	0.10	60.02	0.602	653	1363	302	0.02	0.17	0.0004	.14e06	10.96	0.039	0.90	33.34
14.09	14.60	21.64	78.8	8.84	0.10	60.10	0.690	657	1355	302	0.04	0.19	0.0003	.13e06	10.77	0.044	0.90	35.61
14.39	14.65	22.56	81.5	8.93	0.06	59.42	0.630	692	1362	302	0.01	0.01	0.0003	.13e06	10.79	0.015	0.90	32.13
14.36	14.17	22.85	88.9	8.88	0.10	60.33	0.757	693	1355	302	0.01	0.02	0.0003	.14e06	11.00	0.015	0.90	40.26
14.75	14.71	23.88	93.1	8.69	0.13	59.65	0.690	696	1374	302	0.01	0.01	0.0003	.13e06	11.14	0.015	0.90	46.53
14.60	14.50	23.17	92.6	8.35	0.16	59.10	0.692	691	1365	302	0.01	0.01	0.0002	.14e06	11.03	0.015	0.90	46.84
14.55	14.03	23.31	95.4	8.76	0.19	59.24	0.660	692	1355	302	0.01	0.01	0.0004	.14e06	11.35	0.015	0.90	51.29
14.95	14.12	22.95	90.1	8.60	0.23	59.27	0.641	694	1360	302	0.01	0.01	0:0003	.13e06	10.86	0.015	0.90	47.56

Table A.14: Results obtained for the multi-objective function by SA (for dimension of cast billet 100 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.013	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.018	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.023	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.028	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.033	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.038	1.20	61.86
14.86	14.86	25.17	112.3	10.7	0.09	67.05	0.789	400	1369	302	.2e-8	0.50	0.0008	.91e05	8.17	0.043	1.20	61.86
12.09	12.09	24.14	84.99	12.0	0.07	50.27	0.531	513	1378	302	0.28	0.39	0.0005	.90e05	8.23	0.014	1.20	43.22
15.28	15.28	27.82	120.0	10.7	0.10	53.62	0.236	400	1389	302	0.31	0.50	0.0004	.90e05	8.27	0.010	1.20	44.76
16.29	16.29	27.43	83.47	10.4	0.19	59.93	0.604	400	1363	302	0.26	0.41	0.0007	.92e05	8.19	0.017	1.20	46.72
19.38	19.36	27.03	97.26	8.44	0.24	60.02	0.788	400	1379	303	.2e-8	0.50	0.0006	.92e05	6.25	0.017	1.20	61.86
15.31	15.31	21.89	119.7	6.05	0.27	50.00	0.504	400	1358	303	0.22	0.28	0.0009	.91e05	6.62	0.021	1.20	47.23
19.17	19.15	28.75	81.80	10.2	0.30	58.01	0.781	400	1333	303	.1e-8	0.50	.4e-5	.89e05	6.02	0.023	1.20	61.83
13.77	13.76	23.79	90.42	11.4	0.36	51.73	0.874	400	1314	303	.2e-7	0.50	0.0002	.88e05	6.00	0.024	1.20	61.86

Table A.15: Results obtained for the multi-objective function by SA (for dimension of cast billet 125 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.010	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.015	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.020	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.025	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.031	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.036	1.20	44.67
14.54	14.54	18.56	112.9	9.38	0.13	53.40	0.690	400	1362	302	0.29	0.41	0.0009	.90e05	8.15	0.040	1.20	44.67
10.97	10.97	21.94	107.2	7.99	0.08	51.45	0.850	400	1370	302	.6e-9	0.50	0.0009	.91e05	8.17	0.010	1.20	61.87
12.80	12.80	26.70	68.67	10.6	0.11	51.42	0.840	400	1377	302	.2e-9	0.50	0.0004	.91e05	8.20	0.022	1.20	61.87
18.44	18.42	25.43	89.84	9.86	0.17	55.34	0.850	400	1412	302	.6e-9	0.50	0.0001	.89e05	8.11	0.016	1.20	61.87
14.15	14.14	29.65	116.5	5.60	0.21	60.56	0.272	400	1375	302	.2e-8	0.50	0.0004	.92e05	8.15	0.015	1.20	61.87
14.97	14.97	21.98	109.8	9.11	0.03	54.34	0.512	411	1387	303	.2e-9	0.50	0.0007	.92e05	6.16	0.019	1.20	61.55
10.20	10.19	25.47	80.12	7.38	0.30	50.51	0.796	400	1331	303	.5e-8	0.50	0.0001	.88e05	5.94	0.018	1.20	61.87
15.46	15.46	29.92	53.01	11.3	0.35	50.00	0.900	579	1284	303	.9e-9	0.50	0.0008	.89e05	5.97	0.026	1.20	56.96

Table A.16: Results obtained for the multi-objective function by SA (for dimension of cast billet 150 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
16.61	16.61	24.09	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0002	.92e05	8.28	0.011	1.20	44.40
16.61	16.61	24.09	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0003	.92e05	8.28	0.016	1.20	44.40
16.61	16.61	24.10	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.30	0.49	0.0002	.92e05	8.28	0.021	1.20	44.40
16.61	16.61	24.10	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0002	.92e05	8.28	0.026	1.20	44.40
16.61	16.61	24.10	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0002	.92e05	8.28	0.031	1.20	44.40
16.61	16.61	24.10	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0003	.92e05	8.28	0.036	1.20	44.40
16.61	16.61	24.10	87.5	8.27	0.11	50.00	0.853	474	1367	302	0.29	0.49	0.0003	.92e05	8.28	0.041	1.20	44.40
10.66	10.66	17.35	116.1	8.72	0.08	53.64	0.894	400	1360	302	.5e-9	0.5	0.0010	.91e05	8.27	0.012	1.20	61.86
16.37	16.37	24.84	118.4	5.18	0.14	57.72	0.590	400	1353	302	0.06	0.50	0.0003	.90e05	8.13	0.016	1.20	58.43
18.96	18.96	24.49	94.06	7.76	0.17	50.60	0.897	400	1367	302	0.08	0.45	0.001	.91e05	8.21	0.027	1.20	56.99
18.18	18.18	18.91	87.86	9.14	0.25	50.00	0.509	556	1365	302	0.41	0.50	0.0008	.92e05	8.24	0.016	1.20	36.61
11.31	11.31	24.38	85.19	11.9	0.26	64.91	0.337	461	1390	302	0.32	0.50	0.0009	.91e05	8.35	0.012	1.20	43.23
18.55	18.55	23.73	89.75	7.10	0.30	62.18	0.848	400	1327	303	.6e-9	0.50	0.0001	.89e05	5.69	0.028	1.20	61.86
14.93	14.92	19.41	94.66	9.77	0.38	52.80	0.715	400	1325	303	.5e-9	0.50	0.0003	.87e05	5.87	0.026	1.20	61.86

Table A.17: Results obtained for the multi-objective function (for dimension of cast billet 100 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
11.90	11.84	28.29	118.9	4.85	0.30	56.00	0.473	400	1387	302	7.e-9	0.50	7.e-9	.14e06	11.7	0.028	1.20	62.14
14.93	14.79	29.97	76.67	6.63	0.34	50.25	0.889	400	1383	302	2.e-9	0.50	0.0009	.14e06	11.8	0.026	1.20	62.14
16.03	16.15	23.68	94.07	11.9	0.30	60.56	0.868	400	1376	302	2.e-9	0.50	0.0003	.14e06	11.6	0.027	1.20	62.14
18.06	17.98	19.58	114.4	11.6	0.32	50.15	0.854	400	1386	302	1.e-9	0.50	0.0009	.14e06	12.0	0.024	1.20	62.14
21.95	21.94	25.16	107.2	8.87	0.34	64.52	0.787	400	1381	302	3.e-9	0.50	0.0005	.14e06	12.0	0.016	1.20	62.14
24.38	24.33	25.52	112.7	7.12	0.30	64.20	0.895	400	1362	302	.9e-9	0.50	0.0006	.15e06	11.8	0.017	1.20	62.14
18.96	19.16	30.00	60.00	11.7	0.30	50.00	0.900	587	1390	302	0.032	0.35	0.0002	.15e06	12.1	0.029	1.20	52.89
15.71	15.87	27.57	79.92	8.34	0.30	50.86	0.682	400	1376	302	0.205	0.50	0.0009	.14e06	11.6	0.015	1.20	51.17
18.46	18.57	29.14	88.08	11.3	0.32	53.75	0.544	400	1375	302	.1e-7	0.50	0.0007	.14e06	11.3	0.015	1.20	62.14
15.61	15.70	18.30	117.8	11.5	0.31	51.65	0.896	400	1354	302	.1e-9	0.50	4.e-5	.14e06	11.2	0.027	1.20	62.14
19.39	19.55	18.23	139.8	11.7	0.34	66.66	0.874	400	1346	302	2.e-9	0.50	0.0005	.15e06	11.3	0.020	1.20	62.14
17.89	17.97	29.67	74.16	9.58	0.33	50.25	0.817	400	1343	302	1.e-8	0.50	0.0009	.14e06	11.4	0.017	1.20	62.14
10.92	11.03	19.00	111.9	10.1	0.30	55.88	0.884	400	1364	302	0.175	0.50	0.0009	.14e06	11.4	0.020	1.20	52.81
31.78	13.82	22.09	106.6	11.3	0.34	62.82	0.872	400	1362	302	1.e-8	0.50	0.0009	.14e06	11.4	0.016	1.20	62.14
18.40	18.25	24.10	100.3	11.4	0.32	69.79	0.892	400	1369	302	.4e-9	0.50	0.0008	.14e06	11.5	0.018	1.20	62.14
17.85	18.03	23.84	99.38	9.76	0.32	70.00	0.899	569	1395	302	0.005	0.50	0.0006	.14e06	11.5	0.010	1.20	56.86
12.56	12.68	10.00	120.0	12.0	0.30	50.00	0.900	691	1347	302	0.490	0.25	0.0005	.15e06	11.6	0.029	1.20	22.18
11.00	11.11	15.00	120.0	12.0	0.30	50.00	0.900	554	1374	302	0.152	0.49	0.0008	.14e06	11.5	0.016	1.20	49.95
18.71	18.90	18.62	119.9	11.8	0.33	50.47	0.898	400	1369	302	1.e-9	0.50	0.0009	.15e06	11.7	0.027	1.20	62.14
17.06	16.98	23.77	93.25	11.2	0.30	56.58	0.891	400	1346	302	2.e-9	0.50	0.0003	.14e06	11.2	0.020	1.20	62.14
11.30	11.38	27.00	99.25	11.7	0.34	63.43	0.714	400	1360	302	2.e-9	0.50	0.0006	.14e06	11.3	0.013	1.20	62.14

Table A.18: Results obtained for the multi-objective function (for dimension of cast billet 125 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
10.70	10.66	29.80	95.65	7.51	0.06	54.60	0.688	400	1325	302	1.e-7	0.50	0.0002	.14e06	11.5	0.019	1.20	64.93
13.66	13.79	19.83	115.2	11.4	0.09	53.18	0.710	400	1363	302	1.e-9	0.50	0.0008	.14e06	11.7	0.023	1.20	64.93
15.08	15.00	21.89	109.2	9.84	0.14	56.33	0.877	400	1362	302	2.e-8	0.50	0.0009	.14e06	11.8	0.015	1.20	64.93
18.18	18.37	28.06	76.38	11.8	0.14	50.96	0.899	400	1390	302	2.e-9	0.50	0.0003	.14e06	11.9	0.018	1.20	64.93
21.96	22.02	29.87	91.10	10.4	0.15	50.04	0.898	400	1376	302	3.e-9	0.50	0.0009	.14e06	11.9	0.017	1.20	64.93
22.86	22.75	29.91	96.22	11.6	0.14	50.21	0.682	400	1395	302	7.e-9	0.50	0.0004	.14e06	11.8	0.027	1.20	64.93
13.53	13.64	29.96	59.94	11.3	0.14	54.42	0.899	400	1401	302	5.e-9	0.50	0.0003	.14e06	11.9	0.011	1.20	64.93
18.44	18.34	27.96	77.99	11.7	0.15	50.63	0.862	400	1389	302	8.e-9	0.50	0.0006	.14e06	11.7	0.011	1.20	64.93
13.52	13.47	28.98	92.79	5.19	0.14	55.67	0.886	400	1368	302	2.e-9	0.50	0.0008	.14e06	11.3	0.028	1.20	64.93
13.78	31.77	25.91	115.4	5.67	0.12	61.71	0.895	400	1372	302	1.e-8	0.50	9.e-5	.14e06	11.3	0.010	1.20	64.93
16.21	16.17	19.26	134.3	10.3	0.14	54.75	0.899	400	1357	302	2.e-9	0.50	0.0003	.14e06	11.1	0.015	1.20	64.93
14.95	15.04	25.10	84.38	10.1	0.14	50.67	0.895	400	1347	302	3.e-9	0.50	0.0001	.14e06	11.3	0.017	1.20	64.93
14.05	14.04	27.91	85.16	9.02	0.14	59.67	0.738	400	1359	302	.4e-9	0.50	0.0002	.15e05	11.3	0.012	1.20	64.93
10.48	10.54	20.63	94.41	10.3	0.12	61.43	0.855	400	1349	302	2.e-9	0.50	0.0005	.15e06	11.8	0.029	1.20	64.93
12.69	12.82	22.61	81.56	11.2	0.13	65.56	0.900	400	1386	302	0.137	0.50	0.0009	.15e06	12.0	0.024	1.20	57.11
12.78	12.91	20.11	84.82	11.8	0.15	70.04	0.900	413	1343	302	0.132	0.50	0.0004	.15e06	11.8	0.017	1.20	53.63
10.00	10.10	9.97	103.6	11.9	0.10	50.19	0.898	405	1368	302	0.044	0.28	0.0007	.15e06	12.0	0.016	1.20	54.95
11.38	11.42	14.56	119.1	11.9	0.14	64.03	0.893	400	1384	302	2.e-9	0.50	0.0002	.15e06	12.1	0.014	1.20	64.93
12.35	12.44	19.72	112.2	10.2	0.09	56.69	0.887	400	1369	302	5.e-9	0.50	0.0004	.15e06	12.0	0.011	1.20	64.93
19.86	19.69	23.47	118.4	11.9	0.15	52.32	0.627	400	1367	302	3.e-9	0.50	0.0009	.15e06	11.9	0.015	1.20	64.93
17.11	17.17	27.76	80.17	10.4	0.15	52.42	0.898	400	1348	302	4.e-9	0.50	0.0003	.15e06	11.5	0.025	1.20	64.93

Table A.19: Results obtained for the multi-objective function (for dimension of cast billet 150 sq.mm.)

Y_P mm	D mm	S mm	f cpm	δ mm	N sec	Sr %	w Kg/t	T_o K	T_s K	T_{Wout} K	ε	δ_1	δ_2	β	ϕ	M m	L m	VEL. mm/sec
11.00	10.93	21.28	86.39	9.14	0.14	60.74	0.883	400	1347	302	8.e-9	0.50	0.0002	.14e06	11.4	0.019	1.20	64.92
13.22	13.13	21.36	71.39	11.9	0.01	50.18	0.889	400	1363	302	1.e-9	0.50	6.e-5	.14e06	11.2	0.025	1.20	64.92
15.98	16.12	25.68	109.9	11.5	0.06	52.69	0.818	400	1379	302	1.e-8	0.50	0.0005	.14e06	11.5	0.027	1.20	64.92
17.50	17.67	25.05	119.1	6.59	0.14	54.56	0.899	400	1361	302	2.e-9	0.50	0.0005	.15e06	11.6	0.020	1.20	64.92
22.06	21.92	26.32	113.6	10.8	0.14	56.35	0.892	400	1359	302	2.e-9	0.50	0.0008	.15e06	11.7	0.019	1.20	64.92
23.05	23.12	27.04	115.8	10.9	0.14	55.80	0.677	400	1377	302	2.e-9	0.50	7.e-5	.14e06	11.7	0.012	1.20	64.92
10.21	10.28	29.93	47.99	10.2	0.05	53.19	0.900	400	1368	302	.1e-9	0.50	0.0009	.15e06	11.9	0.014	1.20	64.12
10.31	10.39	23.27	77.58	8.47	0.13	58.00	0.893	400	1372	302	.5e-8	0.50	0.0009	.14e06	11.7	0.013	1.20	64.92
16.13	16.20	21.35	95.83	11.8	0.12	55.33	0.861	400	1357	302	3.e-9	0.50	0.0002	.15e06	11.4	0.024	1.20	64.92
14.66	14.58	18.57	111.4	10.8	0.14	51.50	0.741	400	1337	302	5.e-9	0.50	0.0006	.15e06	11.3	0.020	1.20	64.92
11.59	11.55	19.58	135.0	6.30	0.11	50.48	0.899	400	1350	302	.3e-8	0.50	0.0001	.15e06	11.0	0.021	1.20	64.92
14.70	14.78	18.03	118.4	11.0	0.14	50.79	0.639	400	1357	302	.5e-9	0.50	0.0004	.14e06	11.4	0.020	1.20	64.92
16.62	16.63	26.88	95.17	9.45	0.13	57.71	0.567	400	1346	302	.4e-9	0.50	0.0006	.14e06	11.2	0.012	1.20	64.92
14.47	14.36	24.47	110.2	9.14	0.09	60.61	0.899	400	1356	302	.1e-7	0.50	0.0004	.14e06	11.4	0.023	1.20	64.92
16.81	16.66	27.16	88.75	11.1	0.10	67.15	0.873	400	1358	302	.1e-8	0.50	0.0004	.15e06	11.4	0.010	1.20	64.92
15.99	15.90	21.69	117.7	10.2	0.14	70.20	0.725	400	1382	302	.8e-9	0.50	0.0005	.14e06	11.1	0.021	1.20	64.92
11.74	11.86	9.899	119.6	11.9	0.06	50.00	0.898	400	1384	302	.1e-9	0.50	0.0006	.15e06	11.8	0.025	1.20	64.92
12.61	12.54	14.14	115.2	11.4	0.12	50.40	0.899	400	1366	302	.8e-8	0.50	0.0002	.15e06	11.6	0.021	1.20	64.92
13.69	13.64	18.75	105.2	11.6	0.10	51.52	0.650	400	1415	302	.2e-8	0.50	0.0003	.15e06	11.7	0.011	1.20	64.92
15.82	15.84	24.51	118.0	8.17	0.11	50.45	0.808	400	1371	302	.2e-8	0.50	0.0004	.15e06	11.4	0.027	1.20	64.92
14.89	14.99	25.60	107.4	9.20	0.09	63.04	0.785	400	1358	302	.5e-8	0.50	0.0006	.15e06	11.3	0.015	1.20	64.92

References

1] Brimacombe J.K., Samarasekera I.V., "The Continuous Casting of Steel", Lecture Notes Presented at Brazilian Steel Industry, Univerisidade Federal Fluminense, Volta Redonda, Brazil, 1995

[2] Brimacombe J.K.,Canadian Metallurgical Quaterly, Vol.15, 1976, pp.163-175

[3] Chakraborti N., Jain D. and Deb K., "Continuous Casting Mold Design Using Genetic Algorithm",(communicated)

[4] Filipic B. and Sarler B., "Evolutionary Optimization of Process Parameters in Continuous Casting of Steel", Design to Manufacture in Modern Industry Proceedings, September 1997, Portoroz, Slovenia

[5] Brimacombe J.K. and Samarasekera I.V., "Fundamental Analysis of the Continuous Casting Process for Quality Improvements", The Centre for Metallurgical Process Engineering, The University of British Columbia, Vancouver, B.C. Canada, V6T 1W5

[6] Samarasekera I.V. and Brimacombe J.K., Ironmaking and Steelmaking, Vol.9, 1982, pp.1-15

[7] Irving W.R., Perkins A. and Gray R., Ironmaking and Steelmaking, Vol.11, 1984, pp.146-151

[8] Irwing W.R.: "Continuous Casting of Steel", The University Press, Cambridge, 1993

[9] Takeuchi E. and Brimacombe J.K.,Metallurgical Transactions B., Vol.15B, 1984, pp.493-509

[10] Grill A., Sorimachi K. and Brimacombe J.K., Metallurgical Transactions, Vol.7B, 1976, pp.176-189

[11] Chakraborti N., Jain D. and Kumar R., "Continuous Casting Mold Design Using Pareto Optimal Genetic Algorithm",(communicated)

[12] Rao S.S.: "Optimization: Theory and Applications", Wiley eastern Limited, NewDelhi, 1978

[13] Nakato H., Sakuraya T., Nozaki T., Emi T. and Nishikawa H., Iron and Steel Society, Warrendale, PA, 1987, pp.23-19

[14] Price K. and Storn R., Dr. Dobbs Journal, Volume22, 1997, pp.18-24

[15] Deb K.: "Optimization for Engineering Design Algorithms and Examples", Prentice Hall India, New-Delhi, 1994

[16] Krishnakumar K., SPIE Vol. 1196, 1989, pp.32-39

[17] Mitchell M.: "An Introduction to Genetic Algorithms", Prentice Hall of India Private. Ltd., USA, 1998

[18] Caroll D., "Developments in Theoretical and Applied Mechanics", Vol.XVIII ed. Wilson H.B., Batra C.W., Bert, A.M., Schapery, T.A., Stewart ,D.S., Swinson, F.F., School of Engineering, University of Alabama, U.S.A, 1996

[19] Goldberg D.E.: "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, Sydney, 1989

[20] Goldberg D.E. and Deb K., TCGA Report No. 90007, Department of Engineering Mechanics, University of Alabama, 1990

[21] Deb K., and Goldberg D.E., "An Investigation of Niche and Species Formation in Genetic Function Optimization", in J.D. Schaffer ed., Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, 1989

[22] Carona A., Marchesi M., Martini C. and Ridella S., ACM Transactions on Mathematical Software, Vol.13, No.3, 1987, pp.262-280

[23] Goffe W.L., Ferrier G.D. and Rogers J., Journal of Econometrics, Vol.60, 1994, pp.65-99

[24] DeJong K. A., "An Analysis of the Behavior of a Class of Genetic Adaptive systems", Doctoral Dissertation, University of Michigan, Dissertation Abstracts International, Vol. 36(10), 5140B, 1975

[25] Deb K., Chakraborti N., "A Combined Heat Transfer and Genetic Algorithm Modeling of an Integrated Steel Plant Bloom Re-heating Furnace", Proceedings 6'th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98), Aachen, Germany, September 7-10, Verlag und Druk Mainz GombH, 1998

[26] Mills K.C., NPL Report DMM(A) 10, Teddington, UK, 1990

[27] Moore J.A., Phillips R.J. and Gibbs T.R., "An Overview for the Requirements of Casting Mold-fluxes", Steel Making Proceedings, ISS-AIME, 1991, pp.615-621

[28] Schrewe H.F., Verlag Stahleisen, Germany, 1987, pp.132-137

[29] Saderman J. and Schrewe H., "The Influence of Casting Powder on the Formation of Cracks in the Continuous Slab Casting", Steel making Conference Proceedings, ISS-AIME, 1991, pp.719-728

[30] Bommaraju R., Iron and Steel Society, Warrendale, PA, 1991, pp.95-110

[31] Koyama K., Nagano K., Nagano Y. and Nakano T., Nippon Steel Technical Report, N.34, 1987, pp.41-47

[32] Sakuraya T., Emi T., Emoto K. and Koshikawa T., "Developments of Mold Fluxes for High Speed Strand Casting Slabs Free from Surface Conditioning", 2nd Process Technology Conference, ISS-AIME, 1981, pp.141-147

[33] Lee I.R., Choi J., Shin K., Kwon O.D. and Choo D.K., "Optimization Technology of Mold Powder According to Casting Conditions", Steel Making Conference Proceeding, ISS-AIME, 1988, pp.175-181

[34] Ogibayashi S., Mukai T., Mimura Y., Nagano Y., Yamaguchi K., Takahashi T., Koyama K. and Koyama T., Nippon Steel Technical Report, N.34, 1987, pp.1-10

[35] Nakano T., Nagano K., Masno N., Fuji M. and Matsuyama T., Nippon Steel Technical Report, N.34, 1987, pp.21-30

[36] Delhale A., Larreq M., Marioton J.F. and Riboud P.V., Warrendale, PA, 1987, pp.15-22

[37] Tsai H. T. and Mastervich J.C., "Influence of Mold Powder on Breakouts at Inland Steel's No.2 Slab Caster", Steel Making Conference Proceeding, ISS-AIME, 1988, pp.167-173

[38] Harris D.J., Otterman B.A., Sellers B.T. and Young J.D., "Development of Casting Practice for Defect Free Steel", Steel Making Conference Proceeding, ISS-AIME, 1987, pp.145-152

[39] Continuous Casting of Steel 1985 - A Second Survey, IISI, Brussels, Belgium, 1986, pp.2.21-2.31

[40] Zasowski P.J. and Sosinski D.J., "Control of Heat Removal in the Continuous Casting Mould", Steel Making Conference Proceeding, ISS-AIME, 1990, pp.253-259

[41] Emling W.H. and Dawson S., "Mold Instrumentation for Breakouts Detection and Control", Steel Making Conference Proceeding, ISS-AIME, 1991, pp.197-217

[42] Mahapatra R.B., Samarasekera I.V. and Brimacombe J.K., Metallurgical Transactions B, Vol 22B, 1991, pp.875-888

[43] Mills K.C., Grieveson P., Olusanya A. and Bagha B., Continuous Casting'85, The Metals Society, London, 1985

[44] Pinherio C.A., Samarasekera I.V. and Brimacombe J.K., "Mold Flux for Continuous Casting of Steel", The Center for Metallurgical Process Engineering, UBC, Canada

[45] Nakato H., Ozawa M., Kinoshita K., Habu Y. and Emi T., Tetsu-to-Hagane, 67, 1981, No.8 and Trans ISIJ, 1984, pp.957-965

[46] Wolf M.M., "Investigation into the Relationship between Heat Flux and Shell Growth in Continuous Casting Moulds", 97th ISIJ Meeting, April 1979, ISIJ, Vol.20, 1980, pp.710-717

[47] Mahapatra R.B., Brimacombe J.K., Samarasekera I.V., Waker N., Paterson E.A. and Young J.D., "Mold Behavior and its Influence on Quality in the Continuous Casting of Steel: PartI. Industrial Trials Mold Temperature Measurements, and Mathematical Modeling", UBC, Canada.

[48] Szekely J. and Themelis N.J.: "Rate Phenomena in Process Metallurgy", Wiley-Inter Science, New York, NY, 1971

[49] Geiger G.H. and Poirier D.E.: "Transport Phenomena in Metallurgy", Addison-Wesley Reading, MA, USA, 1973

[50] Arora A.C. and Domkundwar S.: "Heat and Mass Transfer", Dhanpat Rai & sons, New Delhi, India, 1992

[51] Brimacombe J.K. and Samarasekera I.V., "Pri'nci'pios do Lingotamento Conti'nuo do Aco", Fluminense Federal University (UFF), Brazil, 1995